34th Computational Complexity Conference

CCC 2019, July 18–20, 2019, New Brunswick, NJ, USA

Edited by

Amir Shpilka

Editors

Amir Shpilka
Tel Aviv University, Tel Aviv, 69978, Israel
shpilka@tauex.tau.ac.il

ACM Classification 2012
Theory of computation

ISBN 978-3-95977-116-0

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/978-3-95977-116-0.

Publication date
July, 2019

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.2019.0

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

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Preface

The papers in this volume were accepted for presentation at the 34th Computational Complexity Conference (CCC 2019), held July 18-–20, 2019 in New Brunswick, New Jersey. 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). CCC 2019 was co-organized by the Center for Discrete Mathematics and Theoretical Computer Science (DIMACS) and co-sponsored by Microsoft Research.

The call for papers sought original research papers in all areas of computational complexity theory. Of the 99 submissions the program committee selected 32 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 and especially its president Venkatesan Guruswami, past president Dieter van Melkebeek and secretary Ashwin Nayak for their advice and assistance; Ran Raz and Rocco Servedio for sharing their knowledge as prior PC chairs for CCC; the Local Arrangements Committee chair Eric Allender; Dor Minzer and Ran Raz for their invited talks; and Michael Wagner for coordinating the production of these proceedings.

Amir Shpilka
Program Committee Chair, on behalf of the Program Committee

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
COMPUTATIONAL COMPLEXITY CONFERENCE

Awards

The program committee of the 34th Computational Complexity Conference is very pleased to present the **Best Student Paper Award** to Josh Alman for his paper

Limits on the Universal Method for Matrix Multiplication.

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
COMPUTATIONAL COMPLEXITY CONFERENCE

Conference Organization

Program Committee

Andrej Bogdanov, Chinese University of Hong Kong
Irit Dinur, Weizmann Institute of Science
Yuval Filmus, Technion - Israel Institute of Technology
Pavel Hrubeš, Czech Academy of Sciences
Valentine Kabanets, Simon Fraser University
Gillat Kol, Princeton University
Troy Lee, University of Technology Sydney
Raghu Meka, University of California at Los Angeles
Ramprasad Saptharishi, Tata Institute of Fundamental Research
Amir Shpilka (Chair), Tel Aviv University
Madhu Sudan, Harvard University

Local Arrangements Committee

Eric Allender (Chair), Rutgers
Tamra Carpenter, Rutgers
Swastik Kopparty, Rutgers
Periklis A. Papakonstantinou, Rutgers
Michael Saks, Rutgers
Shubhangi Saraf, Rutgers

Board of Trustees

Boaz Barak, Harvard University
Sevag Gharibian, University of Paderborn and Virginia Commonwealth University
Venkatesan Guruswami (President), Carnegie Mellon University
Shachar Lovett, University of California at San Diego
Dieter van Melkebeek, University of Wisconsin-Madison
Ashwin Nayak, University of Waterloo
Ryan O'Donnell, Carnegie Mellon University
Rahul Santhanam, Oxford University
Rocco Servedio, Columbia University
Ronen Shaltiel, University of Haifa

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
COMPUTATIONAL COMPLEXITY CONFERENCE

External Reviewers

Dorit Aharonov
Andris Ambainis
Boaz Barak
Siddharth Bhandari
Marco Carmosino
Lijie Chen
Amin Coja-Oghlan
Daniel Dadush
Dean Doron
Frederik Garbe
Parikshit Gopalan
Tom Gur
Matthew Hastings
Edward A. Hirsch
Rahul Jain
Benny Kimelfeld
Robin Kothari
Joseph M. Landsberg
Yi Li
Inbal Livni Navon
Nikhil Mande
Or Meir
Huy Nguyen
Igor Carboni Oliveira
Aduri Pavan
Richard Y. Peng
Youming Qiao
Anup Rao
Maurice J. Rojas
Atri Rudra
Rahul Santhanam
Dominik Scheder
Vatsal Sharan
Makrand Sinha
Noah Stephens-Davidowitz
Avishay Tal
Roei Tell
Madhur Tulsiani
Thomas Vidick
Omri Weinstein
David Woodruff
Jeroen Zuiddam

Eric Allender
Srinivasan Arunachalam
Paul Beame
Amey Bhangale
Arkadev Chattopadhyay
Xi Chen
Roni Con
Anindya De
Aris Filos-Ratsikas
Dmitry Gavinsky
Daniel Gottesman
Tuomas Hakoniemi
Pooya Hatami
Justin Holmgren
Pritish Kamath
Hartmut Klauck
Ashutosh Kumar
Chin Ho Lee
Noam Lifshitz
Zhenjian Lu
Dániel Marx
Tulasimohan Molli
Chinmay Nirkhe
Rafael Oliveira
Chris Peikert
Aaron Potechin
Prasad Raghavendra
David Richerby
Noga Ron-Zewi
Karthik C. S.
Raghuvansh Saxena
Tselil Schramm
Alexander Sherstov
Fang Song
David Steurer
Navid Talebanfard
Neil Thapen
Chris Umans
Marc Vinyals
Christopher Williamson
Jinyu Xie

Josh Alman
Per Austrin
Alexander Belov
Andrei Bulatov
Eshan Chattopadhyay
Gil Cohen
Matthew Coudron
Ilias Diakonikolas
Michael A. Forbes
Alexander Golovnev
Joshua Grochow
Prahladh Harsha
Shuichi Hirahara
Pavel Hubáček
Zander Kelley
Pravesh K. Kothari
Mrinal Kumar
Euiwoong Lee
Nutan Limaye
Meena Mahajan
Andrew McGregor
Dimitrios Myrisiotis
Ryan O'Donnell
Denis Pankratov
Shir Peleg
Pavel Pudlak
Michaël Rao
Robert Robere
Ron Rothblum
Rishi Saket
Ori Sberlo
Ronen Shaltiel
Igor Shinkar
Srikanth Srinivasan
Amnon Ta-Shma
Li-Yang Tan
Luca Trevisan
Vinod Variyam
Ben Lee Volk
Ronald de Wolf
Henry Yuen

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
COMPUTATIONAL COMPLEXITY CONFERENCE

Criticality of Regular Formulas

Benjamin Rossman
Departments of Mathematics and Computer Science, University of Toronto, Canada
ben.rossman@utoronto.ca

Abstract

We define the **criticality** of a boolean function $f : \{0,1\}^n \to \{0,1\}$ as the minimum real number $\lambda \geq 1$ such that

$$\mathbb{P}\left[\ \mathsf{DT}_{\mathsf{depth}}(f\restriction \boldsymbol{R}_p) \geq t\ \right] \leq (p\lambda)^t$$

for all $p \in [0,1]$ and $t \in \mathbb{N}$, where $\boldsymbol{R}_p$ is the p-random restriction and $\mathsf{DT}_{\mathsf{depth}}$ is decision-tree depth. Criticality is a useful parameter: it implies an $O(2^{(1-\frac{1}{2\lambda})n})$ bound on the decision-tree size of f, as well as a $2^{-\Omega(k/\lambda)}$ bound on Fourier weight of f on coefficients of size $\geq k$.

In an unpublished manuscript [11], the author showed that a combination of Håstad's switching and multi-switching lemmas [5, 6] implies that AC^0 circuits of depth $d+1$ and size s have criticality at most $O(\log s)^d$. In the present paper, we establish a stronger $O(\frac{1}{d}\log s)^d$ bound for **regular formulas**: the class of AC^0 formulas in which all gates at any given depth have the same fan-in. This result is based on

(i) a novel switching lemma for bounded size (unbounded width) DNF formulas, and

(ii) an extension of (i) which analyzes a canonical decision tree associated with an entire depth-d formula.

As corollaries of our criticality bound, we obtain an improved #SAT algorithm and tight Linial-Mansour-Nisan Theorem for regular formulas, strengthening previous results for AC^0 circuits due to Impagliazzo, Matthews, Paturi [7] and Tal [17]. As a further corollary, we increase from $o(\frac{\log n}{\log\log n})$ to $o(\log n)$ the number of quantifier alternations for which the QBF-SAT (quantified boolean formula satisfiability) algorithm of Santhanam and Williams [14] beats exhaustive search.

2012 ACM Subject Classification Theory of computation → Circuit complexity

Keywords and phrases AC^0 circuits, formulas, criticality

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.1

Funding *Benjamin Rossman*: NSERC and Sloan Research Fellowship

Acknowledgements I am grateful to Prahladh Harsha, Shrikanth Srinivasan, Siddharth Bhandari, Tulasi Molli and Or Meir for valuable feedback on a preliminary version of this paper.

1 Introduction

For a boolean function f, we consider the random variable $\mathsf{DT}_{\mathsf{depth}}(f\restriction \boldsymbol{R}_p)$ (the decision-tree depth of f under a p-random restriction) parameterized by $p \in [0,1]$. For every f, there is a sufficient small value of $p > 0$ such that $\mathsf{DT}_{\mathsf{depth}}(f\restriction \boldsymbol{R}_p)$ satisfies an exponential tail bound. This "sufficiently small" is quantified by the following notion of *criticality*.

▶ **Definition 1.** *For $\lambda \in \mathbb{R}_{\geq 1}$, we say that a boolean function f is* λ-critical *if*

$$\mathbb{P}\left[\ \mathsf{DT}_{\mathsf{depth}}(f\restriction \boldsymbol{R}_p) \geq t\ \right] \leq (p\lambda)^t$$

for all $p \in [0,1]$ and $t \in \mathbb{N}$. The criticality *of f is the minimum $\lambda \in \mathbb{R}_{\geq 1}$ for which f is λ-critical.*

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka; Article No. 1; pp. 1:1–1:28

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


Criticality has been implicitly studied in previous works, although we are unaware if this parameter of boolean functions has been named before. Most notably, Håstad's switching lemma [5] is equivalent to the statement that every width-w CNF or DNF formula is $O(w)$-critical. Motivating our study of criticality is the observation that upper bounds on criticality imply upper bounds on decision-tree size.

▶ **Theorem 2.** *If $f : \{0,1\}^n \to \{0,1\}$ is λ-critical, then $\mathsf{DT}_{\mathsf{size}}(f)$ is at most $O(2^{(1-\frac{1}{2\lambda})n})$.*

In light of Theorem 2, it is reasonable to expect upper bounds on the criticality of a class of boolean functions to yield (randomized) #SAT algorithms.

In an unpublished manuscript [11], we observed that a combination of Håstad's switching lemma [5] and "multi-swithing lemma" [6] can be used to show that AC^0 circuits of depth $d+1$ and size s have criticality $O(\log s)^d$. Via Theorem 2, this implies an essentially tight upper bound on the decision-tree size of AC^0 circuits and yields a randomized #SAT algorithm with parameters matching that of Impagliazzo, Matthews and Paturi [7] for AC^0 circuits of super-linear size $n^{1+\Omega(1)}$. In the present paper, we improve these results by giving a quantitatively stronger upper bound on the criticality of regular AC^0 formulas, where *regular* means that all gates at the same height have equal fan-in.

▶ **Theorem 3.** *Regular AC^0 formulas of depth $d+1$ and size s have criticality $O(\frac{1}{d}\log s)^d$ (specifically, at most $60^d(\frac{1}{d}\ln s + 1)^d$).*

Theorem 3 unifies (and arguably simplifies) several of the main results on AC^0 circuits, including bounds on decision-tree size and the Fourier spectrum and #SAT algorithms. In addition, by obtaining quantitative stronger versions of these results for regular AC^0 formulas, we improve an algorithm of Santhanam and Williams [14] for satisfiability of quantified boolean formulas with bounded-many quantifier blocks.

1.1 Known bounds on criticality

The following bounds on criticality are immediate or known from previous work.

(1) If f is a boolean function which depends on n variables, then it is n-critical. This follows from

$$\mathbb{P}\left[\ \mathsf{DT}_{\mathsf{depth}}(f{\restriction}\boldsymbol{R}_p) \geq t\ \right] \leq \mathbb{P}\left[\ \mathbf{Bin}(n,p) \geq t\ \right] \leq p^t\binom{n}{t} \leq (pn)^t.$$

(2) If f has decision-tree depth k, then f is k-critical. This follows from the folklore bound: for all $t \geq 1$,

$$\mathbb{P}\left[\ \mathsf{DT}_{\mathsf{depth}}(f{\restriction}\boldsymbol{R}_p) \geq t\ \right] \leq 2^{t-1}p^t\binom{k}{t} \leq \frac{2^{t-1}}{t!}(pk)^t \leq (pk)^t.$$

(The first inequality is shown by induction on t.)

(3) Showing that (1) and (2) are tight: the n-variable parity function has criticality exactly $\lambda = n$. This follows from

$$np(1-p)^{n-1} = \mathbb{P}\left[\ \mathbf{Bin}(n,p) = 1\ \right] \leq \mathbb{P}\left[\ \mathsf{DT}_{\mathsf{depth}}(\mathrm{PARITY}_n{\restriction}\boldsymbol{R}_p) \geq 1\ \right] \leq p\lambda.$$

Therefore, $\lambda \geq n(1-p)^{n-1}$ for all $p \in (0,1]$, hence $\lambda \geq n$.

(4) Håstad's switching lemma [5] shows that every width-w CNF or DNF formula is $O(w)$-critical.

(5) An alternative switching lemma in [11] (included in Section 4 of this paper) shows that every size-m CNF or DNF formula is $O(\log m)$-critical.

(6) By a combination of Hastad's switching lemma [5] and multi-switching lemma [6], it is shown in [11] that every boolean function f computable by an AC^0 circuit of depth $d+1$ and size s is $O(\log s)^d$-critical. The switching lemma is used to show

$$\mathbb{P}\left[\, \mathsf{DT}_{\mathsf{depth}}(f\restriction \boldsymbol{R}_p) \geq t \,\right] \leq \left(p \cdot O(\log s)^d\right)^t$$

for $t \leq \log s$, while the multi-switching lemma establishes this inequality for $t > \log s$.

1.2 Formulas vs. circuits

Every AC^0 circuit of depth $d+1$ and size s is equivalent to a regular AC^0 formula of depth $d+1$ and size at most s^d. Theorem 3 therefore implies the $O(\log s)^d$ criticality bound for AC^0 circuits.

The proof of our quantitatively stronger $O(\frac{1}{d}\log s)^d$ bound for regular AC^0 formulas is based on a novel "depth-d switching lemma". Previous switching lemmas analyze the so-called *canonical decision tree* of bounded-width depth-2 formula under a random restriction. In contrast, we analyze a certain canonical decision tree associated with a depth-d formula under a sequence of random restrictions. The bound we obtain is in terms of *top fan-in*, as opposed to *width* (i.e., bottom fan-in of a depth-2 formula).

While our proof of Theorem 3 relies on the assumption of regularity, we conjecture that the $O(\frac{1}{d}\log s)^d$ criticality bound applies to all AC^0 formulas. In this connection, let us mention that a previous result of the author [12] implies that every boolean function f computed by a (not necessarily regular) AC^0 formula of depth $d+1$ and size s satisfies

$$\mathbb{P}\left[\, \mathsf{DT}_{\mathsf{depth}}(f\restriction \boldsymbol{R}_p) \geq t \,\right] \leq \left(p \cdot O(\tfrac{1}{d}\log s)^d\right)^t$$

for all $t \leq \log s$. The method of [12] applies Håstad's switching lemma to AC^0 formulas in a more efficient way. However, this method encounters the same $\log s$ barrier mentioned in §1.1(6). We do not know how to establish criticality for non-regular AC^0 formulas by proving the above inequality for $t > \log s$.

1.3 Outline of the paper

In Section 2 we state the definitions of AC^0 formulas, restrictions, decision trees, and present some key inequalities. Section 3 gives some results on criticality, including a proof of Theorem 2. In Section 4 we show that size-m DNF formulas have criticality $O(\log m)$ via a novel switching lemma argument. (This section and Appendix A are independent of the rest of the paper, but serve a warm-up for the more complicated switching lemmas that follow.) In Section 5, we introduce a *canonical decision tree* associated with an entire depth-d formula under a chain of restrictions. Sections 6 and 7 prove switching lemmas for this notion of canonical decision tree. Section 8 contains the proof of Theorem 3. Section 9 discusses satisfiability algorithms. The paper concludes with some open questions in Section 10.

2 Preliminaries

$\mathbb{N}$ is the set of natural numbers $\{0, 1, 2, \dots\}$. For $n \in \mathbb{N}$, $[n]$ is the set $\{1, \dots, n\}$ (in particular, $[0]$ is the empty set). $\ln(\cdot)$ is the natural logarithm and $\log(\cdot)$ is the base-2 logarithm. We consistently use boldface for random objects.

Throughout this paper, we fix an arbitrary set V whose elements we call *variable indices*. Without loss of generality, $V = [n]$; however, since the nature and number of variable indices plays no role in our switching lemma, we prefer to think of V as an abstract set. (The only time we assume $V = [n]$ is when speaking of boolean functions $f : \{0,1\}^n \to \{0,1\}$ in §3.)

▶ **Definition 4** (AC^0 formulas). *A* depth-0 formula *is a constant* 0 *or* 1 *or a literal* X_v *or* $\overline{X}_v$ *where* v *is a variable index. For* $d \geq 1$*, a* depth-d formula *is a syntactic object of the form* $\mathrm{OR}(F_1, \ldots, F_m)$ *or* $\mathrm{AND}(F_1, \ldots, F_m)$ *where* $m \geq 1$ *and* $F_1, \ldots, F_m$ *are depth* $d-1$ *formulas.*

We measure size *of a formula by the number of depth-*1 *subformulas. Formally,*

$$\mathrm{size}(F) := \begin{cases} 0 & \text{if } F \text{ has depth } 0, \\ 1 & \text{if } F \text{ has depth } 1, \\ \sum_{\ell=1}^{m} \mathrm{size}(F_\ell) & \text{if } F \text{ has depth} \geq 2 \text{ and is the OR or AND of } F_1, \ldots, F_m. \end{cases}$$

Up to a constant factor, size is equivalent to the number of gates in F*.*

The (syntactic) support *of a formula is the set of variable indices* v *such that the literal* X_v *or* $\overline{X}_v$ *occurs as a depth-*0 *subformula. Throughout this paper, all definitions and proofs* by induction *are, first, with respect to depth, and second, with respect to support size.*

If F *is a formula, we write* $F \equiv 0$ *(resp.* $F \equiv 1$*) if* F *computes the constant* 0 *function (resp. the constant* 1 *function).*

A depth-d *formula is* regular *if there exist integers* $m_2, \ldots, m_d \geq 1$ *such that, for all* $i \in \{2, \ldots, d\}$*, every depth* i *subformula has top fan-in* m_i*. Note that such a formula has size* $\prod_{i=2}^{d} m_i$*.*

▶ **Definition 5** (Restrictions and inputs). *A* restriction *is a partial function* ϱ *from* V *to* $\{0,1\}$*, viewed as a subset of* $V \times \{0,1\}$*, whose elements we denote by* $v \mapsto b$*. We write* $\mathrm{Dom}(\varrho)$ *for the domain of* ϱ*, and we write* $\mathrm{Stars}(\varrho)$ *for the set* $V \setminus \mathrm{Dom}(\varrho)$ *of "unrestricted" variable indices.*

An input *is a restriction with domain* V *(i.e., a total function from* V *to* $\{0,1\}$*, as opposed to a string in* $\{0,1\}^{|V|}$*).*

Two restrictions ϱ *and* σ *are* consistent *(we also say that* σ *is* ϱ-consistent*) if* $\varrho(v) = \sigma(v)$ *for all* $v \in \mathrm{Dom}(\varrho) \cap \mathrm{Dom}(\sigma)$*. In this case, the union* $\varrho \cup \sigma$ *is a restriction. We say that* σ *is a* refinement *of* ϱ *if* $\varrho \subseteq \sigma$ *(i.e.,* σ *extends* ϱ *by fixing additional variables).*

If F *is a formula and* ϱ *is a restriction, we denote by* $F\restriction_\varrho$ *the formula obtained from* F *by relabeling literals according to* ϱ *(we do not perform any simplification to* F*). Formally, we have the induction definition:*

$$F\restriction_\varrho = \begin{cases} F & \text{if } F \text{ is a constant or a literal } X_v \text{ or } \overline{X}_v \text{ where } v \in \mathrm{Stars}(\varrho), \\ 0 & \text{if } F \text{ is } X_v \text{ and } \varrho(v) = 0, \text{ or } F \text{ is } \overline{X}_v \text{ and } \varrho(v) = 1, \\ 1 & \text{if } F \text{ is } X_v \text{ and } \varrho(v) = 1, \text{ or } F \text{ is } \overline{X}_v \text{ and } \varrho(v) = 0, \\ \mathrm{OR/AND}(F_1\restriction_\varrho, \ldots, F_m\restriction_\varrho) & \text{if } F \text{ is } \mathrm{OR/AND}(F_1, \ldots, F_m). \end{cases}$$

Note that the support of $F\restriction_\varrho$ *equals the support of* F *minus the domain of* ϱ*.*

For $p \in [0,1]$*, the* p-random restriction $\boldsymbol{R}_p$ *is the random restriction which independent maps each variable index* v *to* 0 *or* 1 *with probability* $\frac{1-p}{2}$*, or leaves* v *unrestricted with probability* p*. For a restriction* ϱ*, a* p-random refinement *of* ϱ *is the random restriction* $\boldsymbol{R}_p$ *conditioned on being an extension of* ϱ *(i.e., conditioned on* $\varrho \subseteq \boldsymbol{R}_p$*).*

▶ **Definition 6** (Sequences and bitstrings). *For integers $s, t \in \mathbb{N}$ and arbitrary sequences $\alpha = \langle \alpha_1, \ldots, \alpha_s \rangle$ and $\beta = \langle \beta_1, \ldots, \beta_t \rangle$, we write $\alpha \circ \beta$ for the concatenated sequence $\langle \alpha_1, \ldots, \alpha_s, \beta_1, \ldots, \beta_t \rangle$. The unique sequence of length 0 is denoted by $\langle\rangle$. (Note that $\langle\rangle$ is the identity with respect to concatenation.)*

We refer to sequences $a = \langle a_1, \ldots, a_s \rangle$ in the set $\{0,1\}^s$ as bitstrings. *(To avoid confusion, we regard inputs to formulas as total functions $V \to \{0,1\}$ rather than as ordered bitstrings in the set $\{0,1\}^{|V|}$.) For $t \geq s$, we write $\{0,1\}^t_s$ for the set of bitstrings $q = \langle q_1, \ldots, q_t \rangle \in \{0,1\}^t$ such that $q_1 + \cdots + q_t = s$. For bitstrings $a \in \{0,1\}^s$ and $b \in \{0,1\}^t$ and $q \in \{0,1\}^t_s$, we write $b \leftarrow^q a$ for the bitstring $\langle c_1, \ldots, c_t \rangle$ defined by*

$$c_j := \begin{cases} b_j & \text{if } q_j = 0, \\ a_i & \text{if } q_j = 1 \text{ and } q_1 + \cdots + q_{j-1} = i \text{ (i.e., } q_j \text{ is the } i^{th} \text{ 1-coordinate of } q). \end{cases}$$

That is, $b \leftarrow^q a$ overwrites b with a in the indices specified by q.

▶ **Definition 7** (Ordered restrictions). *An* ordered restriction *is a sequence $\beta = \langle v_1 \mapsto b_1, \ldots, v_t \mapsto b_t \rangle$ where $t \in \mathbb{N}$ and each $v_i \mapsto b_i$ is an ordered pairs with $v_i \in V$ and $b_i \in \{0,1\}$ such that $v_1, \ldots, v_t$ are distinct. As a matter of notation, we sometimes identify β with its underlying (unordered) restriction $\{v_1 \mapsto b_1, \ldots, v_t \mapsto b_t\}$, for instance, by writing $\mathrm{Dom}(\beta)$ for $\{v_1, \ldots, v_t\}$ or $F|_\beta$ for $F|_{\{v_1 \mapsto b_1, \ldots, v_t \mapsto b_t\}}$.*

For an ordered restriction $\beta = \langle v_1 \mapsto b_1, \ldots, v_t \mapsto b_t \rangle$ and a set of variable indices $S \subseteq V$ and a bitstring $a = \langle a_1, \ldots, a_s \rangle \in \{0,1\}^{|\mathrm{Dom}(\beta) \cap S|}$, we write $\beta \leftarrow_S a$ for the ordered restriction $\langle v_1 \mapsto c_1, \ldots, v_t \mapsto c_t \rangle$ where

$$c_j := \begin{cases} b_j & \text{if } v_j \notin S, \\ a_i & \text{if } v_j \text{ is the } i^{th} \text{ variable of } \mathrm{Dom}(\beta) \cap S \text{ in the order given by } \beta. \end{cases}$$

In other words, $\langle c_1, \ldots, c_t \rangle = \langle b_1, \ldots, b_t \rangle \leftarrow^q a$ where $q \in \{0,1\}^t_s$ is the bitstring defined by $q_j = 1 :\Leftrightarrow v_j \in S$.

▶ **Definition 8** (Decision trees). *A* decision tree *is a finite rooted binary tree T in which each non-leaf is labeled by a variable index v and the two edges to its children are labeled by 0 and 1. We require the variable indices on any root-to-leaf branch are distinct; each root-to-leaf branch therefore corresponds to an ordered restriction. We measure* size *by the number of leaves and* depth *by the maximum number of non-leaves on a root-to-leaf path.*

We say that a decision tree T determines *a boolean function f if the restricted function $f|_\alpha$ is constant for each ordered restriction α corresponding to a branch of T. (We might also say that T "computes" f, if we regard T as having output values on leaves.) The* decision-tree size *(resp.* decision-tree depth*) of a boolean function f, denoted $\mathsf{DT}_{\mathsf{size}}(f)$ (resp. $\mathsf{DT}_{\mathsf{depth}}(f)$), is the minimum size (resp. depth) of a decision tree that determines f.*

Later on, it will be convenient to identify decision trees with the set of ordered restrictions corresponding to branches. From this perspective, a decision tree is a nonempty set T^* of ordered restrictions such that, for all $\langle v_1 \mapsto a_1, \ldots, v_s \mapsto a_s \rangle \in T^*$ and $i \in [s]$,

- *if $\langle v_1 \mapsto a_1, \ldots, v_{i-1} \mapsto a_{i-1}, v'_i \mapsto a'_i \rangle$ is an initial subsequence of any element of T^*, then $v'_i = v_i$,*
- *$\langle v_1 \mapsto a_1, \ldots, v_{i-1} \mapsto a_{i-1}, v_i \mapsto 1 - a_i \rangle$ is an initial subsequence of some element of T^*.*

2.1 Inequalities

▶ **Lemma 9.** *For every integer $s \geq 1$ and $\varepsilon \in (0,1]$,*

$$\sum_{t=s}^{\infty}(1-\varepsilon)^t\binom{t-1}{s-1} = \left(\frac{1-\varepsilon}{\varepsilon}\right)^s.$$

Proof. Let $\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots$ be independent Bernoulli(ε) random variables. Then

$$1 = \mathbb{P}\left[\sum_{i=1}^{\infty}\boldsymbol{X}_i \geq s\right] = \sum_{t=s}^{\infty}\mathbb{P}\left[\boldsymbol{X}_t = 1 \text{ and } \sum_{i=1}^{t-1}\boldsymbol{X}_i = s-1\right] = \sum_{t=s}^{\infty}\varepsilon^s(1-\varepsilon)^{t-s}\binom{t-1}{s-1}.$$

The identity follows by multiplying both sides by $((1-\varepsilon)/\varepsilon)^s$. ◀

The next inequality also comes up in the $\mathrm{AC}^0[\oplus]$ formula lower bound of Rossman and Srinivasan [13].

▶ **Lemma 10.** *For all real numbers $a, b, c \geq 0$,*

$$\left(\frac{a}{c}+1\right)^c(b+1) \leq \left(\frac{a+b}{c+1}+1\right)^{c+1}.$$

Proof. The lemma is trivial if $c = 0$ (under the convention that $(\frac{a}{0}+1)^0 = 1$), so assume $c > 0$. Let $f(a,b,c) := \mathrm{RHS} - \mathrm{LHS}$. Then

$$\frac{\partial}{\partial b}f(a,b,c) = \left(\frac{a+b}{c+1}+1\right)^c - \left(\frac{a}{c}+1\right)^c.$$

Note that this is an increasing function of b with a zero at $b = a/c$. Therefore, $f(a,b,c)$ is minimal at $b = a/c$ where it takes value $f(a, a/c, c) = 0$. ◀

As a corollary, we get:

▶ **Lemma 11.** *For all integers $d, m_1, \ldots, m_d \geq 1$,*

$$\prod_{i=1}^{d}(\ln m_i + 1) \leq \left(\frac{1}{d}\ln\left(\prod_{i=1}^{d} m_i\right) + 1\right)^d.$$

Proof. For each $j \in [d-1]$, Lemma 10 implies

$$\left(\frac{1}{j-1}\ln\left(\prod_{i=1}^{j-1} m_i\right) + 1\right)^{j-1}(\ln m_j + 1) \leq \left(\frac{1}{j}\ln\left(\prod_{i=1}^{j} m_i\right) + 1\right)^j.$$

The lemma follows from these $d-1$ inequalities. ◀

The final inequality of this section plays a key role in our switching lemma analysis.

▶ **Lemma 12.** *Let I be a finite set and let $\mu : I \to [0,1]$ be a function such that $\sum_{i\in I}\mu(i) \leq 1$. Then for every function $t : I \to \mathbb{R}_{\geq 1}$ and $s \in \mathbb{R}_{\geq 1}$,*

$$\sum_{i\in I}\left(\frac{t(i)}{s}\right)^s\mu(i) \leq \left(\frac{1}{s}\ln\left(\sum_{i\in I}e^{t(i)}\mu(i)\right) + 1\right)^s.$$

Proof. Let $\lambda := \sum_{i\in I} \mu(i)$ ($\in [0,1]$) and let $\nu(i) := \mu(i)/\lambda$ ($\geq \mu(i)$). We have

$$\sum_{i\in I} \left(\frac{t(i)}{s}\right)^s \mu(i) = \lambda \sum_{i\in I} \left(\frac{t(i)}{s}\right)^s \nu(i) \leq \lambda \sum_{i\in I} \left(\frac{1}{s}\ln(e^{t(i)}) + 1\right)^s \nu(i)$$
$$\leq \lambda \left(\frac{1}{s}\ln\left(\sum_{i\in I} e^{t(i)}\nu(i)\right) + 1\right)^s$$

by Jensen's inequality since $a \mapsto (\frac{1}{s}\ln(a)+1)^s$ is concave over $a \in \mathbb{R}_{\geq 1}$

$$= \lambda \left(\frac{1}{s}\ln\left(\sum_{i\in I} e^{t(i)}\mu(i)\right) + 1 - \frac{1}{s}\ln(\lambda)\right)^s$$
$$\leq \left(\frac{1}{s}\ln\left(\sum_{i\in I} e^{t(i)}\mu(i)\right) + 1\right)^s$$

since $\lambda \mapsto \lambda(a - \frac{1}{s}\ln(\lambda))^s$ is increasing over $\lambda \in (0,1]$ (hence maximal at $\lambda = 1$) for every $a \in \mathbb{R}_{\geq 1}$. ◀

In the special case $t(i) = \ln(1/\mu(i))$ and $s = 1$, we get the inequality $\sum_{i\in I} \mu(i)\ln(1/\mu(i)) \leq \ln|I| + 1$. For distributions μ, this is essentially the inequality $\mathbb{H}(\mu) \leq \log|\mathrm{Support}(\mu)|$ for Shannon entropy; when μ is a sub-distribution, this inequality requires $+ O(1)$ on the righthand side.

3 Implications of Criticality

Before presenting our main result on regular AC^0 formulas, we give some general results on λ-critical functions. We begin with the following upper bound on decision-tree size, which is slightly stronger than Theorem 2.

▶ **Proposition 13.** *If $f : \{0,1\}^n \to \{0,1\}$ is $\frac{1}{2\varepsilon}$-critical, then* $\mathsf{DT}_{\mathsf{size}}(f) \leq 2^{n-\varepsilon n-\sqrt{\varepsilon n}+\log(\varepsilon n)+O(1)}$.

Proof. We first note that the proposition is trivial if $\varepsilon > \frac{1}{2}$, since no non-constant boolean function has criticality < 1. If $n < 10$ or $\varepsilon < \frac{2}{n}$, then the bound $\mathsf{DT}_{\mathsf{size}}(f) \leq 2^{n-\varepsilon n-\sqrt{\varepsilon n}+\log(\varepsilon n)+O(1)}$ follows from the trivial bound $\mathsf{DT}_{\mathsf{size}}(f) \leq 2^n$ by choosing a large enough constant. We may therefore assume that $n \geq 10$ and $\varepsilon \in [\frac{2}{n}, \frac{1}{2}]$.

Let $p := \varepsilon - \frac{1}{n}$ and note that $p \in [\frac{1}{n}, \frac{1}{2}]$. We have

$$\mathbb{E}\left[\,\mathsf{DT}_{\mathsf{size}}(f\restriction \boldsymbol{R}_p)\,\right] \leq \mathbb{E}\left[\,2^{\mathsf{DT}_{\mathsf{depth}}(f\restriction \boldsymbol{R}_p)}\,\right] = \sum_{t=0}^{\infty} 2^t\, \mathbb{P}\left[\,\mathsf{DT}_{\mathsf{depth}}(f\restriction \boldsymbol{R}_p) = t\,\right]$$
$$\leq \sum_{t=0}^{\infty} 2^t \left(\frac{p}{2\varepsilon}\right)^t = \sum_{t=0}^{\infty} \left(1 - \frac{1}{\varepsilon n}\right)^t = \varepsilon n.$$

We next make use of the following facts:

- $\mathbb{P}\left[\,\mathbf{Bin}(n,p) \geq pn + \sqrt{pn+1}\,\right] > 0.05$
  (this bound holds for all $n \geq 10$ and $p \in [\frac{1}{n}, \frac{1}{2}]$, as can be shown using estimates in [19]),
- $\mathsf{DT}_{\mathsf{size}}(f) \leq \sum_{\varrho: S\to\{0,1\}} \mathsf{DT}_{\mathsf{size}}(f\restriction_{\varrho})$ for every set of variable indices $S \subseteq [n]$.

Let $\boldsymbol{S}$ be a p-random subset of $[n]$ (i.e., a uniform random subset of size $\mathbf{Bin}(n,p)$), and let $\boldsymbol{\varrho}$ be a uniform random function $[n] \setminus \boldsymbol{S} \to \{0,1\}$. Note that $\boldsymbol{\varrho}$ is a p-random restriction.

For any $c > 0$, we have

$$\begin{aligned}
&\mathbb{1}[\, \mathsf{DT}_{\mathsf{size}}(f) > c2^{n-\varepsilon n} \,] \\
&\quad \le \mathop{\mathbb{P}}_{\boldsymbol{S}}\left[\, 2^{n-|\boldsymbol{S}|} \mathop{\mathbb{E}}_{\boldsymbol{\varrho}}[\, \mathsf{DT}_{\mathsf{size}}(f\restriction_{\boldsymbol{\varrho}}) \,] > c2^{n-\varepsilon n} \,\right] \\
&\quad \le \mathop{\mathbb{P}}_{\boldsymbol{S}}\left[\, \left(|\boldsymbol{S}| < pn + \sqrt{pn+1} \right) \text{ or } \left(\mathop{\mathbb{E}}_{\boldsymbol{\varrho}}[\, \mathsf{DT}_{\mathsf{size}}(f\restriction_{\boldsymbol{\varrho}}) \,] > c2^{pn+\sqrt{pn+1}-\varepsilon n} \right) \,\right] \\
&\quad \le \mathop{\mathbb{P}}_{\boldsymbol{S}}[\, |\boldsymbol{S}| < pn + \sqrt{pn+1} \,] + \mathop{\mathbb{P}}_{\boldsymbol{S}}\left[\, \mathop{\mathbb{E}}_{\boldsymbol{\varrho}}[\, \mathsf{DT}_{\mathsf{size}}(f\restriction_{\boldsymbol{\varrho}}) \,] > c2^{\sqrt{\varepsilon n}-1} \,\right] \\
&\quad \le \mathbb{P}[\, \mathbf{Bin}(n,p) < pn + \sqrt{pn+1} \,] + \frac{2}{c2^{\sqrt{\varepsilon n}}} \mathbb{E}[\, \mathsf{DT}_{\mathsf{size}}(f\restriction_{\boldsymbol{R}_p}) \,] \\
&\quad < 0.95 + \frac{2\varepsilon n}{c2^{\sqrt{\varepsilon n}}}.
\end{aligned}$$

Setting $c := 40\varepsilon n / 2^{\sqrt{\varepsilon n}}$, we have $\mathbb{1}[\, \mathsf{DT}_{\mathsf{size}}(f) > c2^{n-\varepsilon n} \,] < 1$. We conclude that

$$\mathsf{DT}_{\mathsf{size}}(f) \le c2^{n-\varepsilon n} = 40 \cdot 2^{n-\varepsilon n-\sqrt{\varepsilon n}+\log(\varepsilon n)}. \qquad \blacktriangleleft$$

The following theorem (which includes Theorem 2) lists several consequences of criticality, which follow from Proposition 13 as well as results of Linial, Mansour and Nisan [9] and Tal [17] relating the Fourier spectrum of a boolean function to its degree under a p-random restriction.

▶ **Theorem 14** (Implications of criticality). *If $f : \{0,1\}^n \to \{0,1\}$ is λ-critical, then*

(1) $\mathsf{DT}_{\mathsf{size}}(f) \le O(2^{(1-\frac{1}{2\lambda})n})$,

(2) *f agrees with* PARITY_n *on at most* $\frac{1}{2} + O(2^{-n/2\lambda})$ *fraction of inputs,*

(3) $\mathbb{P}[\, \mathsf{deg}(f\restriction_{\boldsymbol{R}_p}) \ge t \,] \le (p\lambda)^t$ *for all p and t,*

(4) $\sum_{S \subseteq [n] : |S| \ge k} \widehat{f}(S)^2 \le 2e \cdot e^{-k/\lambda}$ *for all k,*

(5) $\sum_{S \subseteq [n] : |S| = k} |\widehat{f}(S)| \le O(\lambda)^k$ *for all k.*

Proof. (1) follows immediately from Proposition 13. Property (2) is a consequence of (1). Property (3) follows from the definition of criticality and the fact that $\mathsf{deg}(\cdot) \le \mathsf{DT}_{\mathsf{depth}}(\cdot)$. Linial, Mansour and Nisan [9] showed that (3) ⇒ (4). Tal [17] showed that (4) ⇒ (5) (and moreover that (4) ⇒ (3), i.e., properties (3) and (4) are equivalent up to constant in the $O(\cdot)$). ◀

We conclude this section by observing that any exponential tail bound on $\mathsf{DT}_{\mathsf{depth}}(f\restriction_{\boldsymbol{R}_q})$ implies an upper bound on criticality.

▶ **Proposition 15.** *Let f be a boolean function, let $q, \varepsilon \in (0,1]$, and suppose $\mathbb{P}[\, \mathsf{DT}_{\mathsf{depth}}(f\restriction_{\boldsymbol{R}_q}) = t \,] \le (1-\varepsilon)^t$ for all $t \in \mathbb{N}$. Then f is $\frac{2}{\varepsilon q}$-critical.*

Proof. Let $0 \le p \le q$, let $\boldsymbol{\varrho}_1$ be a q-random restriction over the variables of f, and let $\boldsymbol{\varrho}_2$ be a p/q-random restriction over $\mathrm{Stars}(\boldsymbol{\varrho}_1)$. Then using §1.1(2) and Lemma 9, we have

$$
\begin{aligned}
&\mathbb{P}\left[\, \mathsf{DT}_{\mathsf{depth}}(f\restriction \boldsymbol{R}_p) \geq t \,\right] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{\varrho}_1}\left[\, \mathop{\mathbb{P}}_{\boldsymbol{\varrho}_2}\left[\, \mathsf{DT}_{\mathsf{depth}}((f\restriction_{\boldsymbol{\varrho}_1})\restriction_{\boldsymbol{\varrho}_2}) \geq t \,\right] \,\right] \\
&= \sum_{k=t}^{\infty} \mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1}\left[\, \mathsf{DT}_{\mathsf{depth}}(f\restriction_{\boldsymbol{\varrho}_1}) = k \,\right] \mathop{\mathbb{E}}_{\boldsymbol{\varrho}_1}\left[\, \mathop{\mathbb{P}}_{\boldsymbol{\varrho}_2}\left[\, \mathsf{DT}_{\mathsf{depth}}((f\restriction_{\boldsymbol{\varrho}_1})\restriction_{\boldsymbol{\varrho}_2}) \geq t \,\right] \,\middle|\, \mathsf{DT}_{\mathsf{depth}}(f\restriction_{\boldsymbol{\varrho}_1}) = k \,\right] \\
&\leq \sum_{k=t}^{\infty} \mathbb{P}\left[\, \mathsf{DT}_{\mathsf{depth}}(f\restriction \boldsymbol{R}_q) = k \,\right] \max_{g\,:\,\mathsf{DT}_{\mathsf{depth}}(g)=k} \mathbb{P}\left[\, \mathsf{DT}_{\mathsf{depth}}(g\restriction \boldsymbol{R}_{p/q}) \geq t \,\right] \\
&\leq \sum_{k=t}^{\infty} (1-\varepsilon)^k \left(\frac{2p}{q}\right)^t \binom{k}{t} = \left(\frac{2p}{q}\right)^t \frac{(1-\varepsilon)^{t-1}}{\varepsilon^t} \leq \left(\frac{2p}{\varepsilon q}\right)^t .
\end{aligned}
$$

◀

4 Criticality of Size-m DNF Formulas

In this section, we show that size-m (unbounded width) DNF formulas have criticality $O(\log m)$ via a novel switching lemma. This switching lemma is based on convexity (it uses the inequality in Lemma 12). As a simple illustration of the underlying idea, in Appendix A we present a simple entropy argument showing that size-m DNF formulas have average sensitivity $O(\log m)$.

The switching lemma for DNF formulas in this section serves as a warm-up for more complicated switching lemmas for (sequences of) depth-d formulas in Sections 6 and 7. Those switching lemmas analyze a different construction of canonical decision trees. (Our result for DNF formulas is technically distinct from the depth-2 case of our depth-d switching lemma.)

Let us now fix a DNF formula $F = \mathrm{OR}(F_1, \ldots, F_m)$ where each term F_ℓ is an AND of literals. We identify each F_ℓ with an ordered restriction $\beta_\ell = \langle v_1 \mapsto b_1, \ldots, v_t \mapsto b_t \rangle$ corresponding to its unique minimal satisfying assignment, and we let $V_\ell = \mathrm{Dom}(\beta_\ell) = \{v_1, \ldots, v_t\}$. We say that a restriction ϱ *satisfies* F_ℓ if $\beta_\ell \subseteq \varrho$, and we say that ϱ *falsifies* F_ℓ if there exists $v \in V_\ell \cap \mathrm{Dom}(\varrho)$ such that $\beta_\ell(v) \neq \beta_\ell(\varrho)$.

For restrictions ϱ, we define the *canonical decision tree* $\mathcal{CDT}(F, \varrho)$ inductively as follows:

- If ϱ satisfies F_ℓ for any $\ell \in [m]$, or if ϱ falsifies F_ℓ for every $\ell \in [m]$, then $\mathcal{CDT}(F)$ is the trivial decision tree $\{\langle\rangle\}$.
- Otherwise, let $\ell \in [m]$ be the unique index such that ϱ falsifies $F_1, \ldots, F_{\ell-1}$ but not F_ℓ. Let $Q := V_\ell \cap \mathrm{Stars}(\varrho)$ and note that $|Q| \geq 1$. In this case, $\mathcal{CDT}(F, \varrho)$ queries all variables in Q, receives answers $\alpha : Q \to \{0,1\}$, and then proceeds as the decision tree $\mathcal{CDT}(F, \varrho \cup \alpha)$.

  Formally, if $\beta_\ell = \langle v_1 \mapsto b_1, \ldots, v_t \mapsto b_t \rangle$ and $Q = \{v_{i_1}, \ldots, v_{i_s}\}$ where $1 \leq i_1 < \cdots < i_s \leq t$, then we have

$$
\begin{aligned}
\mathcal{CDT}(F, \varrho \cup \alpha) := \{(v_{i_1} \mapsto a_1, \ldots, v_{i_s} \mapsto a_s) \circ \beta :& \\
a \in \{0,1\}^s,\ \beta \in \mathcal{CDT}(F, \varrho \cup \{v_{i_1} \mapsto a_1, \ldots, v_{i_s} \mapsto a_s\})\}&.
\end{aligned}
$$

Note that the decision tree $\mathcal{CDT}(F, \varrho)$ determines the function computed by $F\restriction_\varrho$.

▶ **Lemma 16.** *Suppose $\mathcal{CDT}(F, \varrho)$ has depth $s \geq 1$. Then there exist*

- *integers $r \in [s]$ and $s_1, \ldots, s_r \geq 1$ with $s_1 + \cdots + s_r = s$,*
- *integers $1 \leq \ell_1 < \cdots < \ell_r \leq m$,*
- *sets $Q_i \subseteq V_{\ell_i} \setminus (V_{\ell_1} \cup \cdots \cup V_{\ell_{i-1}})$ with $|Q_i| = s_i$ and restrictions $\alpha_i, \sigma_i : Q_i \to \{0,1\}$ for each $i \in [r]$*

such that, for all $i \in [r]$,

(i) $\varrho \cup \alpha_1 \cup \cdots \cup \alpha_{i-1}$ *falsifies* $F_{\ell'}$ *for all* $1 \le \ell' < \ell_i$,

(ii) $\varrho \cup \alpha_1 \cup \cdots \cup \alpha_{i-1} \cup \sigma_i$ *satisfies* F_{ℓ_i},

(iii) $Q_i = (V_{\ell_i} \setminus (V_{\ell_1} \cup \cdots \cup V_{\ell_{i-1}})) \cap \mathrm{Stars}(\varrho)$.

Proof. Straightforward from unpacking the inductive definition of $\mathcal{CDT}(F, \varrho)$. ◀

▶ **Lemma 17.** *Let* $F = \mathrm{OR}(F_1, \ldots, F_m)$ *be a DNF formula and let* $\boldsymbol{\varrho}$ *be a p-random restriction. Then*

$$\mathbb{P}[\ \mathcal{CDT}(F, \boldsymbol{\varrho}) \text{ has depth } s\ ] \le (8ep\log(em))^s.$$

Proof. By Lemma 16, we have

$$\begin{aligned}
&\mathop{\mathbb{P}}_{\boldsymbol{\varrho}}[\ \mathcal{CDT}(F, \boldsymbol{\varrho}) \text{ has depth } s\ ] \\
&\quad\le \sum_{r, s_1, \ldots, s_r, \ell_1, \ldots, \ell_r, Q_1, \ldots, Q_r, \alpha_1, \ldots, \alpha_r, \sigma_1, \ldots, \sigma_r} \mathop{\mathbb{P}}_{\boldsymbol{\varrho}}[\ \text{(i),(ii),(iii) for all } i \in [r]\ ] \\
&\quad\le 2^s \max_{r, \vec{s}} \sum_{\vec{\ell}, \vec{Q}, \vec{\alpha}, \vec{\sigma}} \mathop{\mathbb{P}}_{\boldsymbol{\varrho}}[\ \text{(i),(ii),(iii) for all } i \in [r]\ ].
\end{aligned}$$

The second inequality uses the fact that there are at most 2^s possibilities for data $(r, s_1, \ldots, s_r)$.

Let $\boldsymbol{x} : V \to \{0,1\}$ be a uniform random completion of $\boldsymbol{\varrho}$. For any restriction γ, let $\boldsymbol{x}^\gamma$ be the input where $\boldsymbol{x}^\gamma(v)$ equals $\gamma(v)$ if $v \in \mathrm{Dom}(\gamma)$ and $x(v)$ otherwise. For any $r, \vec{s}, \vec{\ell}, \vec{Q}, \vec{\alpha}, \vec{\sigma}$, note that

$$\begin{aligned}
\mathop{\mathbb{P}}_{\boldsymbol{\varrho}}[\ \text{(i),(ii),(iii) for all } i \in [r]\ ] &= 2^s \mathop{\mathbb{P}}_{\boldsymbol{\varrho}, \boldsymbol{x}}[\ \text{(i),(ii),(iii) for all } i \in [r] \text{ and } \sigma_1 \cup \cdots \cup \sigma_r \subseteq \boldsymbol{x}\ ] \\
&= 2^s \mathop{\mathbb{P}}_{\boldsymbol{\varrho}, \boldsymbol{x}}[\ \text{(i'),(ii'),(iii') for all } i \in [r]\ ] \\
&= (2p)^s (1-p)^{|V_1 \cup \cdots \cup V_{\ell_r}| - s} \mathop{\mathbb{P}}_{\boldsymbol{x}}[\ \text{(i'),(ii') for all } i \in [r]\ ] \\
&\le (2p)^s \mathop{\mathbb{P}}_{\boldsymbol{x}}[\ \text{(i'),(ii') for all } i \in [r]\ ]
\end{aligned}$$

where

(i') $\boldsymbol{x}^{\alpha_1 \cup \cdots \cup \alpha_{i-1}}$ falsifies $F_{\ell'}$ and $1 \le \ell' < \ell_i$,

(ii') $\boldsymbol{x}^{\alpha_1 \cup \cdots \cup \alpha_{i-1}}$ satisfies F_{ℓ_i},

(iii') $Q_i = (V_{\ell_i} \setminus (V_{\ell_1} \cup \cdots \cup V_{\ell_{i-1}})) \cap \mathrm{Stars}(\boldsymbol{\varrho})$.

Letting

$$\mu(\vec{\ell}, \vec{Q}, \vec{\alpha}) := \mathop{\mathbb{P}}_{\boldsymbol{x}}[\ \text{(i'),(ii') for all } i \in [r]\ ],$$

we have

$$\mathop{\mathbb{P}}_{\boldsymbol{\varrho}}[\ \mathcal{CDT}(F, \boldsymbol{\varrho}) \text{ has depth } t\ ] \le (4p)^s \max_{r, \vec{s}} \sum_{\vec{\ell}, \vec{Q}, \vec{\alpha}} \mu(\vec{\ell}, \vec{Q}, \vec{\alpha}).$$

We next observe that, given any $\ell_1, \ldots, \ell_i$, there are $2^{s_i} \binom{|V_{\ell_i} \setminus (V_{\ell_1} \cup \cdots \cup V_{\ell_{i-1}})|}{s_i}$ choices

for (Q_i, α_i). Therefore,

$$\sum_{\vec{\ell},\vec{Q},\vec{\alpha}} \mu(\vec{\ell},\vec{Q},\vec{\alpha}) \le 2^s \sum_{\ell_1} \max_{Q_1,\alpha_1} \binom{|V_{\ell_1}|}{s_1} \sum_{\ell_2} \max_{Q_2,\alpha_2} \binom{|V_{\ell_2} \setminus V_{\ell_1}|}{s_2} \cdots$$
$$\cdots \sum_{\ell_r} \max_{Q_r,\alpha_r} \binom{|V_{\ell_2} \setminus (V_{\ell_1} \cup \cdots \cup V_{\ell_{r-1}})|}{s_r} \mu(\vec{\ell},\vec{Q},\vec{\alpha})$$
$$\le 2^s \sum_{\ell_1} \max_{Q_1,\alpha_1} \sum_{\ell_2} \max_{Q_2,\alpha_2} \sum_{\ell_r} \max_{Q_r,\alpha_r} \binom{|V_{\ell_1} \cup \cdots \cup V_{\ell_r}|}{s} \mu(\vec{\ell},\vec{Q},\vec{\alpha}).$$

We replace each $\sum \max$ with $\max \sum$ as follows. Let $Q_i^\star$ and $\alpha_i^\star$ range over functions $Q_i^\star(\ell_1,\dots,\ell_i) \in \binom{V_{\ell_i} \setminus (V_{\ell_1} \cup \cdots \cup V_{\ell_{i-1}})}{s_i}$ and $\alpha_i^\star(\ell_1,\dots,\ell_i) : Q_i^\star(\ell_1,\dots,\ell_i) \to \{0,1\}$, and let $\mu(\vec{\ell},\vec{Q}^\star,\vec{\alpha}^\star)$ be short for

$$\mu(\langle \ell_1,\dots,\ell_r\rangle, \langle Q_1^\star(\ell_1),\dots,Q_r^\star(\ell_1,\dots,\ell_r)\rangle, \langle \alpha_1^\star(\ell_1),\dots,\alpha_r^\star(\ell_1,\dots,\ell_r)\rangle).$$

This allows us replace each $\sum_{\ell_1,\dots,\ell_i} \max_{Q_i,\alpha_i}$ with $\max_{Q_i^\star,\alpha_i^\star} \sum_{\ell_1,\dots,\ell_i}$ to obtain

$$\sum_{\vec{\ell},\vec{Q},\vec{\alpha}} \mu(\vec{\ell},\vec{Q},\vec{\alpha}) \le 2^s \max_{\vec{Q}^\star,\vec{\alpha}^\star} \sum_{\vec{\ell}} \binom{|V_{\ell_1} \cup \cdots \cup V_{\ell_r}|}{s} \mu(\vec{\ell},\vec{Q}^\star,\vec{\alpha}^\star).$$

A key observation is that, for any given $\vec{Q}^\star$ and $\vec{\alpha}^\star$, we have $\sum_{\vec{\ell}} \mu(\vec{\ell},\vec{Q}^\star,\vec{\alpha}^\star) \le 1$. To see why, note that each input x determines at most one sequence $\vec{\ell} = \langle \ell_1,\dots,\ell_r\rangle$ such that (i') and (ii') hold for all $i \in [r]$, that is, $x^{\alpha_1^\star(\ell_1) \cup \cdots \cup \alpha_{i-1}^\star(\ell_1,\dots,\ell_{i-1})}$ satisfies F_{ℓ_i} and falsifies $F_{\ell'}$ for all $\ell' < \ell_i$. Therefore, the events (over random $\boldsymbol{x}$) defining probabilities $\mu(\vec{\ell},\vec{Q}^\star,\vec{\alpha}^\star)$ are mutually exclusive. We now have the following bound, using Lemma 12 for the last inequality:

$$\sum_{\vec{\ell}} \binom{|V_{\ell_1} \cup \cdots \cup V_{\ell_r}|}{s} \mu(\vec{\ell},\vec{Q}^\star,\vec{\alpha}^\star) \le \sum_{\vec{\ell}} \left(\frac{e|V_{\ell_1} \cup \cdots \cup V_{\ell_r}|}{s}\right)^s \mu(\vec{\ell},\vec{Q}^\star,\vec{\alpha}^\star)$$
$$= \left(\frac{e}{\ln(2)}\right)^s \sum_{\vec{\ell}} \left(\frac{\ln(2^{|V_{\ell_1} \cup \cdots \cup V_{\ell_r}|})}{s}\right)^s \mu(\vec{\ell},\vec{Q}^\star,\vec{\alpha}^\star)$$
$$\le \left(\frac{e}{\ln(2)}\right)^s \left(\frac{1}{s} \ln\left(\sum_{\vec{\ell}} 2^{|V_{\ell_1} \cup \cdots \cup V_{\ell_r}|} \mu(\vec{\ell},\vec{Q}^\star,\vec{\alpha}^\star)\right) + 1\right)^s.$$

A second key observation is that, for any $\vec{\ell},\vec{Q},\vec{\alpha}$, we have

$$\mu(\vec{\ell},\vec{Q},\vec{\alpha}) = \mathbb{P}[\text{ (i'),(ii') for all } i \in [r]\ ]$$
$$\le \mathbb{P}[\text{ (ii') for all } i \in [r]\ ]$$
$$= \begin{cases} (1/2)^{|V_{\ell_1} \cup \cdots \cup V_{\ell_r}|} & \text{if } \bigwedge_{i \in [r]} F_{\ell_i}\restriction_{\alpha_i \cup \cdots \cup \alpha_{i-1}} \text{ is satisfiable,} \\ 0 & \text{otherwise.} \end{cases}$$

Therefore,

$$\sum_{\vec{\ell}} 2^{|V_{\ell_1} \cup \cdots \cup V_{\ell_r}|} \mu(\vec{\ell},\vec{Q}^\star,\vec{\alpha}^\star) \le \sum_{\vec{\ell}} 1 \le \binom{m}{r} \le m^r \le m^s.$$

Putting the pieces together, we conclude

$$\mathbb{P}_{\boldsymbol{\varrho}}[\ \mathcal{CDT}(F,\boldsymbol{\varrho}) \text{ has depth } s\ ] \le \left(\frac{8ep}{\ln(2)}\right)^s \left(\frac{1}{s}\ln(m^s) + 1\right)^s = \left(8ep\log(em)\right)^s. \qquad \blacktriangleleft$$

▶ **Corollary 18.** *Every size-m DNF formula has criticality at most* $16e\log(em)$.

Proof. Let F be a size-m DNF formula. Without loss of generality, let $t \geq 1$ and $0 < p \leq (16e\log(em))^{-1}$. Then

$$\begin{aligned}\mathop{\mathbb{P}}_{\boldsymbol{\varrho}}[\ \mathsf{DT}_{\mathsf{depth}}(F\restriction_{\boldsymbol{\varrho}}) \geq t\ ] &\leq \sum_{s=t}^{\infty} \mathop{\mathbb{P}}_{\boldsymbol{\varrho}}[\ \mathcal{CDT}(F,\boldsymbol{\varrho}) \text{ has depth } s\ ] \\ &\leq \sum_{s=t}^{\infty} \Big(8ep\log(em)\Big)^{s} \leq \Big(16ep\log(em)\Big)^{t}.\end{aligned}$$

◀

5 The Canonical Decision Tree of a Depth-d Formula

In this section, we define the canonical decision tree of a depth-d formula F under a chain of restrictions $\varrho_1 \subseteq \cdots \subseteq \varrho_d$, denoted $\mathcal{T}^*_{\varrho_1,\ldots,\varrho_d}(F)$. The primary definition, however, is of a richer object $\mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F)$.

▶ **Definition 19.** *For every $d \in \mathbb{N}$ and chain of restrictions $\varrho_1 \subseteq \cdots \subseteq \varrho_d$ and depth-d formula F, we define a set of ordered restrictions $\mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F)$ as follows. In the base case $d = 0$, let*

$$\mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F) := \begin{cases}\{\langle\rangle\} & \text{if } F \text{ is a constant } 0 \text{ or } 1,\\ \{\langle v \mapsto 0\rangle, \langle v \mapsto 1\rangle\} & \text{if } F \text{ is a literal } X_v \text{ or } \overline{X}_v.\end{cases}$$

For $d \geq 1$, the definition is inductive. Suppose $F = \mathrm{OR}(F_1,\ldots,F_m)$ where each F_ℓ is a depth $d-1$ formula. Assume that $T_{\varrho_1,\ldots,\varrho_{d-1}}(F_\ell)$ is defined for all $\ell \in [m]$ and that $T_{\varrho_1,\ldots,\varrho_d}(F\restriction_\gamma)$ is defined for every restriction γ whose domain includes at least one variable index in the support of F. We consider three cases:

- **(i)** *If $F_1\restriction_{\varrho_d} \equiv \cdots \equiv F_m\restriction_{\varrho_d} \equiv 0$, then $\mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F) := \{\langle\rangle\}$.*
- **(ii)** *If $F_1\restriction_{\varrho_d} \equiv \cdots \equiv F_{\ell-1}\restriction_{\varrho_d} \equiv 0$ and $F_\ell\restriction_{\varrho_d} \equiv 1$, then $\mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F) := \{\langle\rangle\}$.*
- **(iii)** *If $F_1\restriction_{\varrho_d} \equiv \cdots \equiv F_{\ell-1}\restriction_{\varrho_d} \equiv 0$ and $F_\ell\restriction_{\varrho_d}$ computes a non-constant function, then $\mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F)$ is the set of ordered restrictions $\langle v_1 \mapsto a_1, \ldots, v_u \mapsto a_u\rangle$ of length $u \geq 1$ such that there exist $t \in [u]$ and $b \in \{0,1\}^t$ satisfying*
  - *$\langle v_1 \mapsto b_1, \ldots, v_t \mapsto b_t\rangle \in \mathcal{T}_{\varrho_1,\ldots,\varrho_{d-1}}(F_\ell)$,*
  - *$\langle v_{t+1} \mapsto a_{t+1}, \ldots, v_u \mapsto a_u\rangle \in \mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F\restriction_{\{v_1\mapsto a_1,\ldots,v_t\mapsto a_t\}})$, and*
  - *for all $i \in [t]$, if $v_i \in \mathrm{Dom}(\varrho_d)$, then $b_i = a_i = \varrho_d(v_i)$; and if $v_i \in \mathrm{Stars}(\varrho_d)$, then*

$$b_i = \begin{cases} a_i & \text{if } (F_\ell\restriction_{\varrho_d})\restriction_{\{v_1\mapsto b_1,\ldots,v_{i-1}\mapsto b_{i-1},\, v_i\mapsto a_i\}} \not\equiv 0,\\ 1-a_i & \text{if } (F_\ell\restriction_{\varrho_d})\restriction_{\{v_1\mapsto b_1,\ldots,v_{i-1}\mapsto b_{i-1},\, v_i\mapsto a_i\}} \equiv 0.\end{cases}$$

Finally, $\mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F)$ is defined in the same way if $F = \mathrm{AND}(F_1,\ldots,F_m)$, but with the roles 0 and 1 exchanged.

▶ **Lemma 20.** *$\mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F)$ is nonempty and every $\alpha = \langle v_1 \mapsto a_1, \ldots, v_u \mapsto a_u\rangle \in \mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F)$ satisfies:*

- **(a)** *α is consistent with ϱ_d (i.e., for all $i \in [u]$, if $v_i \in \mathrm{Dom}(\varrho_d)$, then $a_i = \varrho_d(v_i)$),*
- **(b)** *the support of F contains $\mathrm{Dom}(\alpha)$ (i.e., for all $i \in [u]$, the literal X_{v_i} or $\overline{X}_{v_i}$ occurs as a depth-0 subformula of F),*
- **(c)** *for all $i \in [u]$, if $v_i \in \mathrm{Stars}(\varrho_d)$, then $\langle v_1 \mapsto a_1, \ldots, v_{i-1} \mapsto a_{i-1}, v_i \mapsto 1 - a_i\rangle$ is an initial subsequence of some element of $\mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F)$,*

(d) *for all $i \in [u]$ and every variable index v'_i and bit $a'_i \in \{0,1\}$, if $\langle v_1 \mapsto a_1, \ldots, v_{i-1} \mapsto a_{i-1}, v'_i \mapsto a'_i\rangle$ is an initial subsequence of any element of $\mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F)$, then $v'_i = v_i$,*

(e) *the function computed by $F\restriction_{\varrho_d \cup \alpha}$ is constant (i.e., $F\restriction_{\varrho_d\cup\alpha} \equiv 0$ or $F\restriction_{\varrho_d\cup\alpha} \equiv 1$).*

Proof. Though the proof is straightforward from Definition 19, we include full details. Note that the lemma is trivial when $d = 0$ as well as in cases (i) and (ii) when $d \geq 1$. So we assume that $F = \mathrm{OR}(F_1, \ldots, F_m)$ falls under case (iii), as witnessed by $\ell \in [m]$. By the induction hypothesis, we may assume that the lemma holds with respect to F_ℓ as well as $F\restriction_\gamma$ for every restriction γ whose domain includes at least one variable index in the support of F.

We first establish that $\mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F)$ is nonempty. By the induction hypothesis, $\mathcal{T}_{\varrho_1,\ldots,\varrho_{d-1}}(F_\ell)$ is nonempty. Since $F_\ell\restriction_{\varrho_d}$ is non-constant and $F_\ell\restriction_{\varrho_d\cup\beta}$ is constant for every $\beta \in \mathcal{T}_{\varrho_1,\ldots,\varrho_{d-1}}(F_\ell)$, there exists $\beta \in \mathcal{T}_{\varrho_1,\ldots,\varrho_{d-1}}(F_\ell)$ such that $F_\ell\restriction_{\varrho_d\cup\beta} \equiv 1$. Let $\beta = \langle v_1 \mapsto b_1, \ldots, v_t \mapsto b_t\rangle$ and note that $t \geq 1$. For all $i \in [t]$, we have $(F_\ell\restriction_{\varrho_d})\restriction_{\{v_1\mapsto b_1,\ldots,v_i\mapsto b_i\}} \not\equiv 0$, since this is the same formula as $F_\ell\restriction_{\varrho_d\cup\{v_1\mapsto b_1,\ldots,v_i\mapsto b_i\}}$ by ϱ_d-consistency of β. By the definition of $\mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F)$ in case (iii), it follows that $\beta\circ\gamma \in \mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F)$ for every $\gamma \in \mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F\restriction_\beta)$. Nonemptiness of $\mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F)$ therefore follows from nonemptiness of $\mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F\restriction_\beta)$, which we know by the induction hypothesis applied to $F\restriction_\beta$ (noting that $\mathrm{Dom}(\beta)$ contains a variable index in the support of F, namely v_1, which is in the support of F_ℓ by the induction hypothesis applied to F_ℓ).

Now consider any $\alpha = \langle v_1 \mapsto a_1, \ldots, v_u \mapsto a_u\rangle \in \mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F)$, as witnessed by some $t \in [u]$ and $b \in \{0,1\}^t$ in definition of case (iii). Let $\gamma := \{v_1 \mapsto a_1, \ldots, v_t \mapsto a_t\}$. We establish properties (a)–(e) in order.

(a): Suppose $i \in [u]$ and $v_i \in \mathrm{Dom}(\varrho_d)$. If $i \in \{1,\ldots,t\}$, then $a_i = \varrho_d(v_i)$ by definition of $\mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F)$ in case (iii). Otherwise, if $i \in \{t+1,\ldots,u\}$, then $a_i = \varrho_d(v_i)$ by property (a) with respect to $F\restriction_\gamma$.

(b): From the induction hypothesis, we know that $v_1, \ldots, v_t$ are in the support of F_ℓ (hence also the support of F) and that $v_{t+1}, \ldots, v_t$ are in the support of $F\restriction_\gamma$ (hence also the support of F).

(c): First note that $(F_\ell\restriction_{\varrho_d})\restriction_{\{v_1\mapsto b_1,\ldots,v_{i-1}\mapsto b_{i-1}\}} \not\equiv 0$ for all $i \in [t]$, as easily shown by induction on i. It then follows from the definition of case (iii) that for all $i \in [t]$, if $v_i \in \mathrm{Stars}(\varrho_d)$, then $\langle v_1 \mapsto a_1, \ldots, v_{i-1} \mapsto a_{i-1}, v_i \mapsto 1 - a_i\rangle$ is an initial subsequence of some element of $\mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F)$. The same conclusion for all $i \in \{t+1,\ldots,u\}$ follows from property (c) with respect to $F\restriction_\gamma$.

(d): If $i \in [t]$ and $\langle v_1 \mapsto a_1, \ldots, v_{i-1} \mapsto a_{i-1}, v'_i \mapsto a'_i\rangle$ is an initial subsequence of an element of $\mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F)$, then by definition of case (iii), $\langle v_1 \mapsto b_1, \ldots, v_{i-1} \mapsto b_{i-1}, v'_i \mapsto a'_i\rangle$ is initial subsequence of an element of $\mathcal{T}_{\varrho_1,\ldots,\varrho_{d-1}}(F_\ell)$ and therefore $v'_i = v_i$. For $i \in \{t+1,\ldots,u\}$, the conclusion follows from property (d) with respect to $F\restriction_\gamma$.

(e): Since $\langle v_{t+1} \mapsto a_{t+1}, \ldots, v_u \mapsto a_u\rangle \in \mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F\restriction_\gamma)$, the formula $(F\restriction_\gamma)\restriction_{\varrho_d\cup\{v_{t+1}\mapsto a_{t+1},\ldots,v_u\mapsto a_u\}}$, which is the same formula as $F\restriction_{\varrho_d\cup\alpha}$ by ϱ_d-consistency of α, computes a constant function by property (e) with respect to $F\restriction_\gamma$. ◀

▶ **Definition 21.** *For $\alpha = \langle v_1 \mapsto a_1, \ldots, v_u \mapsto a_u\rangle \in \mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F)$, let α^* denote the subsequence*

$$\alpha^* := \langle v_{i_1} \mapsto a_{i_1}, \ldots, v_{i_s} \mapsto a_{i_s}\rangle$$

for the unique $1 \leq i_1 < \cdots < i_s \leq u$ such that $\{v_{i_1}, \ldots, v_{i_s}\} = \mathrm{Dom}(\alpha) \cap \mathrm{Stars}(\varrho_d)$. Let

$$\mathcal{T}^*_{\varrho_1,\ldots,\varrho_d}(F) := \{\alpha^* : \alpha \in \mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F)\}.$$

▶ **Lemma 22.** *$\mathcal{T}^*_{\varrho_1,\ldots,\varrho_d}(F)$ is the set of branches of a decision tree determining $F\restriction_{\varrho_d}$. Moreover, each element of $\mathcal{T}^*_{\varrho_1,\ldots,\varrho_d}(F)$ is a subsequence of a unique element of $\mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F)$.*

Proof. Straightforward from Lemma 20. ◀

▶ **Definition 23.** *We call $\mathcal{T}^*_{\varrho_1,\ldots,\varrho_d}(F)$ the* canonical decision tree of $F\restriction_{\varrho_d}$ under $\varrho_1,\ldots,\varrho_d$. *For a bitstring $a \in \{0,1\}^s$ and an ordered restriction α, we write "$\mathcal{T}^{(a)}_{\varrho_1,\ldots,\varrho_d}(F) = \alpha$" if $\alpha \in \mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F)$ and there exist variable indices $v_1,\ldots,v_s$ such that $\alpha^* = \langle v_1 \mapsto a_1, \ldots, v_s \mapsto a_s\rangle$. We say that "$\mathcal{T}^{(a)}_{\varrho_1,\ldots,\varrho_d}(F)$ exists" if $\mathcal{T}^{(a)}_{\varrho_1,\ldots,\varrho_d}(F) = \alpha$ for any $\alpha \in \mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F)$.*

Note that, by Lemma 22, if $\mathcal{T}^{(a)}_{\varrho_1,\ldots,\varrho_d}(F)$ exists then $\mathcal{T}^{(a)}_{\varrho_1,\ldots,\varrho_d}(F) = \alpha$ for a unique $\alpha \in \mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F)$ (justifying our use of the equality symbol). We may regard $\mathcal{T}^{(\cdot)}_{\varrho_1,\ldots,\varrho_d}(F)$ as a partial function from bitstrings to elements of the set $\mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F)$.

6 Depth-d Switching Lemma

In this section, we consider a depth-d formula $F = \mathrm{OR}(F_1,\ldots,F_m)$ and study the branches of $\mathcal{T}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_d}(F)$ where $\boldsymbol{\varrho}_1 \subseteq \cdots \subseteq \boldsymbol{\varrho}_d$ is a chain of random restrictions where $\boldsymbol{\varrho}_d$ is a p-random refinement of $\boldsymbol{\varrho}_{d-1}$ and formulas $F_1,\ldots,F_m$ satisfy a certain hypothesis with respect to $\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_{d-1}$. This allows us to bound the probability that $\mathcal{T}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_d}(F)$ has an a-branch for any string $a \in \{0,1\}^s$. We refer to the main result of this section, Proposition 25, as the "depth-d switching lemma" since it analyzes the canonical decision tree of F in a similar manner as Håstad's switching lemma analyzes the canonical decision tree of a CNF or DNF formula.

Proposition 25 is in fact a special case of the slightly more general Proposition 27 ("serial depth-d switching lemma"), which we prove in the next section. The proofs are essentially the same, but with Proposition 25 we have fewer indices to keep track of. The next lemma unpacks the recursive definition of $\mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F)$ to obtain a more explicit characterization of its branches. This lemma associates $\mathcal{T}^{(a)}_{\varrho_1,\ldots,\varrho_d}(F)$, whenever this exists, with certain data $(r,\vec{s},\vec{\ell},\vec{t},\vec{b},\vec{q})$.

▶ **Lemma 24** (Unpacking $\mathcal{T}^{(a)}_{\varrho_1,\ldots,\varrho_d}(F)$)**.** *Let $F = \mathrm{OR}(F_1,\ldots,F_m)$ be a depth-d formula, let $\varrho_1 \subseteq \cdots \subseteq \varrho_d$ be restrictions, let $s \geq 1$, and let $a \in \{0,1\}^s$. If $\mathcal{T}^{(a)}_{\varrho_1,\ldots,\varrho_d}(F)$ exists, then there exist*

- *integers $r \in [s]$ and $s_1,\ldots,s_r \geq 1$ with $s_1 + \cdots + s_r = s$,*
- *integers $1 \leq \ell_1 < \cdots < \ell_r \leq m$,*
- *integers $t_i \geq s_i$ and bitstrings $b_i \in \{0,1\}^{t_i}$ and $q_i \in \{0,1\}^{t_i}_{s_i}$ for each $i \in [r]$*

with the property that there exist unique ordered restrictions $\beta_1,\ldots,\beta_r$ such that, for all $i \in [r]$,

- **(i)** *$(F_{\ell'}\restriction_{\gamma_i})\restriction_{\varrho_d} \equiv 0$ for all $1 \leq \ell' < \ell_i$,*
- **(ii)** *$(F_{\ell_i}\restriction_{\gamma_i})\restriction_{\varrho_d} \not\equiv 0$,*
- **(iii)** *$\mathcal{T}^{(b_i)}_{\varrho_1,\ldots,\varrho_{d-1}}(F_{\ell_i}\restriction_{\gamma_i}) = \beta_i$,*
- **(iv)** *β_i is ϱ_d-consistent and $(F_{\ell_i}\restriction_{\gamma_i})\restriction_{\varrho_{d-1}\cup\beta_i} \equiv 1$,*
- **(v)** *"q_i identifies $\mathrm{Stars}(\varrho_d)$ within $\mathrm{Dom}(\beta_i) \cap \mathrm{Stars}(\varrho_{d-1})$" in the following sense: for all $j \in [t_i]$,*

$$q_{i,j} = 1 \iff \mathrm{Stars}(\varrho_d) \text{ contains the } j^{th} \text{ variable of } \mathrm{Dom}(\beta_i) \cap \mathrm{Stars}(\varrho_{d-1}) \text{ in the order given by } \beta_i.$$

where
$$\gamma_i := (\beta_1 \circ \cdots \circ \beta_{i-1}) \leftarrow_{\mathrm{Stars}(\varrho_{d-1})} c_i,$$
$$c_i := (\, b_1 \circ \cdots \circ b_{i-1}\,) \leftarrow^{q_1 \circ \cdots \circ q_{i-1}} \langle a_1, \ldots, a_{s_1+\cdots+s_{i-1}} \rangle.$$

(Note that conditions (iii) and (v) imply that $|\mathrm{Dom}(\beta_i) \cap \mathrm{Stars}(\varrho_{d-1})| = t_i$ and $|\mathrm{Dom}(\beta_i) \cap \mathrm{Stars}(\varrho_d)| = s_i$ and $\gamma_i = (\beta_1 \circ \cdots \circ \beta_{i-1}) \leftarrow_{\mathrm{Stars}(\varrho_d)} \langle a_1, \ldots, a_{s_1+\cdots+s_{i-1}} \rangle$.)

Proof. Straightforward from Definition 19. ◀

▶ **Proposition 25** ("Depth-d switching lemma"). *Let $F = \mathrm{OR}(F_1, \ldots, F_m)$ be a depth-d formula. Suppose $\boldsymbol{\varrho}_1 \subseteq \cdots \subseteq \boldsymbol{\varrho}_{d-1}$ are random restrictions such that the following holds: for all integers $r \geq 1$ and $1 \leq \ell_1 < \cdots < \ell_r \leq m$ and $t_1, \ldots, t_r \geq 1$ and bitstrings $b_1, \ldots, b_r, c_1, \ldots, c_r$ where $b_i \in \{0,1\}^{t_i}$ and $c_i \in \{0,1\}^{t_1+\cdots+t_{i-1}}$,*

$$\mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_{d-1}} \left[\exists \beta_1, \ldots, \beta_r \bigwedge_{i \in [r]} \left(\mathcal{T}^{(b_i)}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_{d-1}}(F_{\ell_i}\restriction_{(\beta_1 \circ \cdots \circ \beta_{i-1}) \leftarrow_{\mathrm{Stars}(\boldsymbol{\varrho}_{d-1})} c_i}) = \beta_i \right) \right] \leq \left(\frac{1}{2e}\right)^{t_1+\cdots+t_r}.$$

Then for every integer $s \geq 1$ and bitstring $a \in \{0,1\}^s$ and $p \in [0,1]$, letting $\boldsymbol{\varrho}_d$ be a p-random refinement of $\boldsymbol{\varrho}_{d-1}$, we have

$$\mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_d} \left[\mathcal{T}^{(a)}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_d}(F) \text{ exists } \right] \leq \big(4ep(\ln m + 1)\big)^s.$$

Proof. By Lemma 24 and a union bound,

$$\mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_d} \left[\mathcal{T}^{(a)}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_d}(F) \text{ exists } \right]$$
$$\leq \sum_{\substack{r,s_1,\ldots,s_r \\ \ell_1,\ldots,\ell_r \\ t_1,\ldots,t_r \\ b_1,\ldots,b_r \\ q_1,\ldots,q_r}} \mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_d} \left[\begin{array}{l} \exists \beta_1, \ldots, \beta_r \text{ such that, for all } i \in [r], \\ \quad \text{(i)} \ (F_{\ell'}\restriction_{\gamma_i})\restriction_{\boldsymbol{\varrho}_d} \equiv 0 \text{ for all } 1 \leq \ell' < \ell_i \\ \quad \text{(ii)} \ (F_{\ell_i}\restriction_{\gamma_i})\restriction_{\boldsymbol{\varrho}_d} \not\equiv 0 \\ \quad \text{(iii)} \ \mathcal{T}^{(b_i)}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_{d-1}}(F_{\ell_i}\restriction_{\gamma_i}) = \beta_i \\ \quad \text{(iv)} \ \beta_i \text{ is } \boldsymbol{\varrho}_d\text{-consistent and } (F_{\ell_i}\restriction_{\gamma_i})\restriction_{\boldsymbol{\varrho}_{d-1} \cup \beta_i} \equiv 1 \\ \quad \text{(v)} \ q_i \text{ identifies } \mathrm{Stars}(\boldsymbol{\varrho}_d) \text{ within } \mathrm{Dom}(\beta_i) \cap \mathrm{Stars}(\boldsymbol{\varrho}_{d-1}) \\ \text{where } \gamma_i := (\beta_1 \circ \cdots \circ \beta_{i-1}) \leftarrow_{\mathrm{Stars}(\boldsymbol{\varrho}_{d-1})} c_i, \\ \qquad c_i := (\, b_1 \circ \cdots \circ b_{i-1}\,) \leftarrow^{q_1 \circ \cdots \circ q_{i-1}} \langle a_1, \ldots, a_{s_1+\cdots+s_{i-1}} \rangle \end{array} \right]$$
$$\leq 2^s \max_{r,s_1,\ldots,s_r} \sum_{\substack{\ell_1,\ldots,\ell_r,t_1,\ldots,t_r, \\ b_1,\ldots,b_r,q_1,\ldots,q_r}} \mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_d} \left[\exists \beta_1, \ldots, \beta_r \text{ such that (i)–(v) for all } i \in [r] \right].$$

Henceforth, we fix $r, s_1, \ldots, s_r$ and bound the sum over $\vec{\ell}, \vec{t}, \vec{b}, \vec{q}$.

Let $\boldsymbol{x}$ be a uniform random completion of $\boldsymbol{\varrho}_d$. For each choice of $\vec{\ell}, \vec{t}, \vec{b}, \vec{q}$, we have the following key sequence of (in)equalities, which we state below and justify afterwards:

$$
\begin{aligned}
&\mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_d}\Big[\ \exists \beta_1,\ldots,\beta_r \text{ such that (i)–(v) for all } i\in[r]\ \Big] \\
&\qquad = 2^s \mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_d,\boldsymbol{x}}\Big[\ \exists \beta_1,\ldots,\beta_r \text{ such that (i)–(v) and } \beta_i\subseteq \boldsymbol{x} \text{ for all } i\in[r]\ \Big] \\
&\qquad \le 2^s \mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_d,\boldsymbol{x}}\left[\begin{array}{l}
\exists \beta_1,\ldots,\beta_r \text{ such that, for all } i\in[r], \\
\quad \text{(i')}\ \ (F_{\ell'}\restriction_{\gamma_i})(\boldsymbol{x}) = 0 \text{ for all } 1\le \ell' < \ell_i \\
\quad \text{(ii')}\ \ (F_{\ell_i}\restriction_{\gamma_i})(\boldsymbol{x}) = 1 \\
\quad \text{(iii')}\ \ \mathcal{T}^{(b_i)}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_{d-1}}(F_{\ell_i}\restriction_{\gamma_i}) = \beta_i \\
\quad \text{(iv')}\ \ \beta_i \subseteq \boldsymbol{x} \\
\quad \text{(v')}\ \ q_i \text{ identifies } \mathrm{Stars}(\boldsymbol{\varrho}_d) \text{ within } \mathrm{Dom}(\beta_i)\cap \mathrm{Stars}(\boldsymbol{\varrho}_{d-1}) \\
\text{where } \gamma_i := (\beta_1\circ\cdots\circ\beta_{i-1}) \leftarrow_{\mathrm{Stars}(\boldsymbol{\varrho}_{d-1})} c_i, \\
\qquad\qquad c_i := (\,b_1\circ\cdots\circ b_{i-1}\,) \leftarrow^{q_1\circ\cdots\circ q_{i-1}} \langle a_1,\ldots,a_{s_1+\cdots+s_{i-1}}\rangle
\end{array}\right] \\
&\qquad = (2p)^s(1-p)^{(t_1+\cdots+t_r-s)}\mu(\vec{\ell},\vec{t},\vec{b},\vec{q}) \\
&\qquad \le (2p)^s\mu(\vec{\ell},\vec{t},\vec{b},\vec{q})
\end{aligned}
$$

where

$$
\mu(\vec{\ell},\vec{t},\vec{b},\vec{q}) := \mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_d,\boldsymbol{x}}\Big[\ \exists \beta_1,\ldots,\beta_r \text{ such that (i')–(iv') for all } i\in[r]\ \Big].
$$

The first equality follows from the independence of conditions (i)–(v) (which only depend on $\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_d$) and the event that $(\beta_1\cup\cdots\cup\beta_r)\subseteq(\boldsymbol{x}\setminus\varrho_d)$ for any fixed ϱ_d in the support of $\boldsymbol{\varrho}_d$ (this event has probability 2^{-s} since $|\mathrm{Dom}(\beta_i)\cap\mathrm{Stars}(\varrho_d)| = s_i$ for each $i\in[r]$). The subsequent inequality follows from the observation that conditions (i)–(v) together with $(\beta_1\cup\cdots\cup\beta_r)\subseteq\boldsymbol{x}$ imply conditions (i')–(v'). The next equality follows from the independence of conditions (i')–(iv') and condition (v'). To see this, consider the following alternative way of generating $\boldsymbol{\varrho}_d$ and $\boldsymbol{x}$ given $\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_{d-1}$: first generate $\boldsymbol{x}$ as a uniform random completion of $\boldsymbol{\varrho}_{d-1}$ (rather than of $\boldsymbol{\varrho}_d$), then obtain $\boldsymbol{\varrho}_d$ from $\boldsymbol{x}$ by randomly removing each pair $v\mapsto\boldsymbol{x}_v$ with $v\in\mathrm{Stars}(\boldsymbol{\varrho}_{d-1})$ independently with probability $1-p$. The independence of conditions (i')–(iv') and condition (v') is now seen by observing that the former only depends on $\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_{d-1}$ and $\boldsymbol{x}$, while the latter only depends on $\boldsymbol{\varrho}_d$ (for any fixed $\varrho_1,\ldots,\varrho_{d-1},x$ in the support of $\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_{d-1},\boldsymbol{x}$). The probability of the latter event is precisely $p^s(1-p)^{(t_1+\cdots+t_r-s)}$ since for each $i\in[r]$, the set $\mathrm{Dom}(\beta_i)\cap\mathrm{Stars}(\varrho_{d-1})$ contains t_i variables, of which $\mathrm{Stars}(\boldsymbol{\varrho}_d)$ is required to include exactly the s_i variables specified by q_i.

Combining the above inequalities, we have

$$
\mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_d}\big[\ \mathcal{T}^{(a)}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_d}(F) \text{ exists}\ \big] \le (4p)^s \max_{r,s_1,\ldots,s_r}\ \sum_{\ell_1,\ldots,\ell_r,t_1,\ldots,t_r,b_1,\ldots,b_r,q_1,\ldots,q_r} \mu(\vec{\ell},\vec{t},\vec{b},\vec{q}).
$$

We next turn to bounding both the individual probabilities $\mu(\vec{\ell},\vec{t},\vec{b},\vec{q})$ and their sum

$\sum_{\vec{\ell},\vec{t},\vec{b},\vec{q}} \mu(\vec{\ell},\vec{t},\vec{b},\vec{q})$. Ignoring conditions (i') and (ii'), we have

$$\begin{aligned}
&\mu(\vec{\ell},\vec{t},\vec{b},\vec{q}) \\
&\leq \mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_d,\boldsymbol{x}}\left[\ \exists \beta_1,\ldots,\beta_r \text{ such that (iii') and (iv') for all } i\in[r]\ \right] \\
&= \left(\frac{1}{2}\right)^{t_1+\cdots+t_r} \mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_{d-1}}\left[\ \exists \beta_1,\ldots,\beta_r \text{ such that (iii') for all } i\in[r]\ \right] \\
&= \left(\frac{1}{2}\right)^{t_1+\cdots+t_r} \mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_{d-1}}\left[\begin{array}{l}\exists \beta_1,\ldots,\beta_r \text{ such that, for all } i\in[r],\\ \mathcal{T}^{(b_{h,i})}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_{d-1}}(F_{\ell_i}\restriction_{(\beta_1\circ\cdots\circ\beta_{i-1})\leftarrow \mathrm{Stars}(\boldsymbol{\varrho}_{d-1})c_i}) = \beta_i \\ \text{where } c_i := (b_1\circ\cdots\circ b_{i-1}) \leftarrow^{q_1\circ\cdots\circ q_{i-1}} \langle a_1,\ldots,a_{s_1+\cdots+s_{i-1}}\rangle\end{array}\right] \\
&\leq \left(\frac{1}{4e}\right)^{t_1+\cdots+t_r}.
\end{aligned}$$

The first equality follows from independence of conditions (iii') and (iv'). The second equality is a restatement of condition (iii'). The last inequality uses the hypothesis of the theorem concerning formulas $F_1,\ldots,F_m$.

We next bound the sum $\sum_{\vec{\ell},\vec{t},\vec{b},\vec{q}} \mu(\vec{\ell},\vec{t},\vec{b},\vec{q})$. We start out by observing that

$$\sum_{\ell_1,\ldots,\ell_r,t_1,\ldots,t_r,b_1,\ldots,b_r,q_1,\ldots,q_r} \mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_d,\boldsymbol{x}}\left[\ \exists \beta_1,\ldots,\beta_r \text{ such that (i')–(v') for all } i\in[r]\ \right] \leq 1,$$

since these events are mutually exclusive. To see why, consider any $\varrho_1,\ldots,\varrho_d,x$ in the support of $\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_d,\boldsymbol{x}$ and notice that there is a unique process of uniquely determining $\vec{\ell},\vec{t},\vec{b},\vec{q}$ (if any exist) such that conditions (i')–(v') hold. First, we find the unique ℓ_1 (if any exists) such that $F_{\ell_1}(x)=1$ and $F_{\ell'}(x)=0$ for all $1\leq \ell'<\ell_1$ (note that $\gamma_1=\langle\rangle$). Next, let β_1 be the unique branch of $\mathcal{T}_{\varrho_1,\ldots,\varrho_{d-1}}(F_{\ell_1})$ consistent with x, let b_i be the sequence of answers to the queried variable indices on this branch, and let t_i be the length of b_1. If $F_{\ell_1}(x)=0$ or $|\mathrm{Dom}(\beta_1)\cap\mathrm{Stars}(\varrho_d)|\neq s_1$, then the process fails; otherwise, let $q_1\in\{0,1\}^{t_1}_{s_1}$ be the unique bitstring that identifies $\mathrm{Stars}(\varrho_d)$ within $\mathrm{Dom}(\beta_1)\cap\mathrm{Stars}(\varrho_{d-1})$. Having uniquely determined ℓ_1,t_1,b_1,q_1, the process continues by finding the unique ℓ_2 (if any exists) such that $(F_{\ell_2}\restriction_{\gamma_2})(x)=1$ and $(F_{\ell'}\restriction_{\gamma_2})(x)=0$ for all $1\leq\ell'<\ell_1$ (note that γ_2 is completed determined by previous data β_1,b_1,q_1). Continuing in this manner, we find unique t_2,b_2,q_2, etc.

Note that condition (v') uniquely determines bitstrings $q_1,\ldots,q_s$. This condition is omitted from the events in probabilities $\mu(\vec{\ell},\vec{t},\vec{b},\vec{q})$, which therefore are not mutually exclusive as the choice of $q_i\in\{0,1\}^{t_i}_{s_i}$ is now free. However, we can restore mutual exclusivity as follows. For each $i\in[r]$, let $q^\star_i$ range over functions associating each sequence of partial data $(\ell_1,t_1,b_1,\ldots,\ell_i,t_i,b_i)$ with an element $q^\star_i(\ell_1,t_1,b_1,\ldots,\ell_i,t_i,b_i)\in\{0,1\}^{t_i}_{s_i}$. Let

$$\mu(\vec{\ell},\vec{t},\vec{b},\vec{q^\star}) := \mu\Big(\vec{\ell},\vec{t},\vec{b},\langle q^\star_1(\ell_1,t_1,b_1),\ldots,q^\star_r(\ell_1,t_1,b_1,\ldots,\ell_r,t_r,b_r)\rangle\Big).$$

For any choice of $q^\star_1,\ldots,q^\star_r$, the events in probabilities $\mu(\vec{\ell},\vec{t},\vec{b},\vec{q^\star})$ are mutually exclusive over $\vec{\ell},\vec{t},\vec{b}$. (It is a subtle but important point that $q^\star_i$ is a function of $(\ell_1,t_1,b_1,\ldots,\ell_i,t_i,b_i)$ independent of any "future" data ℓ_j,t_j,b_j for $j>i$.) Therefore,

$$\sum_{\ell_1,\ldots,\ell_r,t_1,\ldots,t_r,b_1,\ldots,b_r} \mu(\vec{\ell},\vec{t},\vec{b},\vec{q^\star}) \leq 1.$$

We now have the bound

$$\begin{aligned}
&\sum_{\ell_1,\ldots,\ell_r,t_1,\ldots,t_r,b_1,\ldots,b_r,q_1,\ldots,q_r} \mu(\vec{\ell},\vec{t},\vec{b},\vec{q}) \\
&\qquad \le \sum_{\ell_1,t_1,b_1} \binom{t_1}{s_1} \max_{q_1} \cdots \sum_{\ell_r,t_r,b_r} \binom{t_r}{s_r} \max_{q_r} \mu(\vec{\ell},\vec{t},\vec{b},\vec{q}) \\
&\qquad = \max_{q_1^\star,\ldots,q_r^\star} \sum_{\ell_1,t_1,b_1} \binom{t_1}{s_1} \cdots \sum_{\ell_r,t_r,b_r} \binom{t_r}{s_r} \mu(\vec{\ell},\vec{t},\vec{b},\vec{q}^\star) \\
&\qquad \le \max_{q_1^\star,\ldots,q_r^\star} \sum_{\ell_1,\ldots,\ell_r,t_1,\ldots,t_r,b_1,\ldots,b_r} \binom{t_1+\cdots+t_r}{s} \mu(\vec{\ell},\vec{t},\vec{b},\vec{q}^\star) \\
&\qquad \le \max_{q_1^\star,\ldots,q_r^\star} \sum_{\ell_1,\ldots,\ell_r,t_1,\ldots,t_r,b_1,\ldots,b_r} e^s \left(\frac{t_1+\cdots+t_r}{s}\right)^s \mu(\vec{\ell},\vec{t},\vec{b},\vec{q}^\star) \\
&\qquad \le \max_{q_1^\star,\ldots,q_r^\star} e^s \left(\frac{1}{s} \ln\left(\sum_{\ell_1,\ldots,\ell_r,t_1,\ldots,t_r,b_1,\ldots,b_r} e^{(t_1+\cdots+t_r)} \mu(\vec{\ell},\vec{t},\vec{b},\vec{q}^\star)\right) + 1\right)^s
\end{aligned}$$

where the final inequality is by Lemma 12.

We next have $\sum_{\vec{\ell},\vec{t},\vec{b}} e^{(t_1+\cdots+t_r)} \mu(\vec{\ell},\vec{t},\vec{b},\vec{q}^\star) \le m^s$ as follows:

$$\begin{aligned}
\sum_{\ell_1,\ldots,\ell_r,t_1,\ldots,t_r,b_1,\ldots,b_r} e^{(t_1+\cdots+t_r)} \mu(\vec{\ell},\vec{t},\vec{b},\vec{q}^\star) &\le \sum_{\ell_1,\ldots,\ell_r,t_1,\ldots,t_r,b_1,\ldots,b_r} \left(\frac{1}{4}\right)^{t_1+\cdots+t_r} \\
&\le \sum_{\ell_1,\ldots,\ell_r} \sum_{t=r}^{\infty} \left(\frac{1}{4}\right)^t \sum_{t_1,\ldots,t_r,b_1,\ldots,b_r \,:\, t_1+\cdots+t_r=t} 1 \\
&= \sum_{\ell_1,\ldots,\ell_r} \sum_{t=r}^{\infty} \left(\frac{1}{2}\right)^t \binom{t-1}{r-1} \\
&= \sum_{\ell_1,\ldots,\ell_r} 1 = \binom{m}{r} \le m^r \le m^s.
\end{aligned}$$

The first inequality uses our bound $\mu(\vec{\ell},\vec{t},\vec{b},\vec{q}) \le (1/8e)^{(t_1+\cdots+t_r)}$ (which holds for any $q_1,\ldots,q_r$ including $q_1^\star(\ell_1,t_1,b_1),\ldots,q_r^\star(\ell_1,t_1,b_1,\ldots,\ell_r,t_r,b_r)$). The first equality is due to the fact that there are $\binom{t-1}{r-1}$ choices for integers $t_1,\ldots,t_r \ge 1$ such that $t_1+\cdots+t_r = t$, and there are 2^{t_i} choices for each bitstring $b_i \in \{0,1\}^{t_i}$. The second equality uses Lemma 9. Finally, we use the fact that $r \le s$ since $s_1,\ldots,s_r \ge 1$ are integers such that $s_1+\cdots+s_r = s$.

Putting together these inequalities, we get the desired bound

$$\Pr_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_d} \left[\, \mathcal{T}^{(a)}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_d}(F) \text{ exists } \right] \le \left(4ep(\ln m + 1)\right)^s. \qquad \blacktriangleleft$$

7 Serial Depth-d Switching Lemma

We would like to prove Theorem 3 (our upper bound on the criticality of regular AC^0 formulas) by applying Proposition 25 ("depth-d switching lemma") to each layer of a regular AC^0 formula. Unfortunately, there is a mismatch between the *hypothesis* and the *conclusion* of Proposition 25: the hypothesis applies to a sequence of depth $d-1$ formulas, while the conclusion applies to single depth-d formula (and cannot therefore serve as the hypothesis for a depth $d+1$ formula).

In this section, we prove an extension of Proposition 25 which we require for Theorem 3. We call this result, Proposition 27, the "serial depth-d switching lemma", since it explores the canonical decision trees of a sequence of depth-d formulas $F_1, \dots, F_k$ in order. For integers $s_1, \dots, s_k \geq 1$ and bitstrings $a_h \in \{0,1\}^{s_h}$ $(h \in [k])$, we would like to bound the event that $\mathcal{T}_{\boldsymbol{\varrho}_1,\dots,\boldsymbol{\varrho}_d}(F_1)$ has an a_1-branch, call it α_1, and that $\mathcal{T}_{\boldsymbol{\varrho}_1,\dots,\boldsymbol{\varrho}_d}(F_2{\restriction}_{\alpha_1})$ has an a_2-branch α_2, etc., and finally that $\mathcal{T}_{\boldsymbol{\varrho}_1,\dots,\boldsymbol{\varrho}_d}(F_d{\restriction}_{\alpha_1\circ\cdots\circ\alpha_{k-1}})$ has an a_k-branch α_k. However, in order for the conclusion of Proposition 27 to match the hypothesis, we need to consider a more general event where, instead of considering $\mathcal{T}_{\boldsymbol{\varrho}_1,\dots,\boldsymbol{\varrho}_d}(F_h{\restriction}_{\alpha_1\circ\cdots\circ\alpha_{h-1}})$ in the hth stage, we instead apply the restriction $(\alpha_1 \circ \cdots \circ \alpha_{h-1}) \leftarrow_{\mathrm{Stars}(\varrho_d)} c_h$ (overwriting $\alpha_1 \circ \cdots \circ \alpha_{h-1}$ on all previously queried variables) where $c_h \in \{0,1\}^{s_1+\cdots+s_{h-1}}$ is an arbitrary bitstring. Although this makes notation in Proposition 27 slightly more cumbersome, the probabilistic main argument is nearly identical to Proposition 25.

▶ **Notation 1.** In what follows, we will consider integers $k \geq 1$ and $r_1, \dots, r_k \geq 1$ and various indexed families $\vec{w} = \{w_{h,i}\}_{h\in[k],i\in[r_h]}$. It is often convenient to regard $\vec{w}$ as a sequence of length $r_1 + \cdots + r_k$:

$$\vec{w} = \langle w_{1,1}, \dots, w_{1,r_1}, \dots\dots, w_{h,1}, \dots, w_{h,r_k}\rangle.$$

For $h \in [k]$ and $i \in [r_k + 1]$, notation "$w_{1,1}, \dots, w_{h,i-1}$" shall refer to the initial subsequence of length $r_1 + \cdots + r_{h-1} + i - 1$:

$$\langle w_{1,1}, \dots, w_{h,i-1}\rangle = \langle w_{1,1}, \dots, w_{1,r_1}, \dots\dots, w_{h-1,1}, \dots, w_{h-1,r_{h-1}}, w_{h,i}, \dots, w_{h,i-1}\rangle.$$

For example, if $w_{h,i}$ are integers (or bitstrings, ordered restrictions, etc.), we will write "$w_{1,1} + \cdots + w_{h,i-1}$" (or "$w_{1,1} \circ \cdots \circ w_{h,i-1}$") for the sum (or composition) of the first $r_1 + \cdots + r_{h-1} + i - 1$ elements of $\vec{w}$.

The following lemma plays the same role in Proposition 27 as Lemma 24 does in Proposition 25.

▶ **Lemma 26.** *Let $F_1, \dots, F_k$ be depth-d formulas where $F_h = \mathrm{OR}(F_{h,1}, \dots, F_{h,m})$ for each $h \in [k]$. Let $\varrho_1 \subseteq \cdots \subseteq \varrho_d$ be restrictions. Let $s_1, \dots, s_k \geq 1$ and let $a_h \in \{0,1\}^{s_h}$ and $c_h \in \{0,1\}^{s_1+\cdots+s_{h-1}}$ for each $h \in [k]$. Suppose there exist ordered restrictions $\alpha_1, \dots, \alpha_k$ such that, for all $h \in [k]$,*

$$\mathcal{T}^{(a_h)}_{\varrho_1,\dots,\varrho_d}(F_h{\restriction}_{(\alpha_1\circ\cdots\circ\alpha_{h-1})\leftarrow_{\mathrm{Stars}(\varrho_d)}c_h}) = \alpha_h.$$

Then there exist

- *integers $r_h \in [s_h]$ and $s_{h,1}, \dots, s_{h,r_h} \geq 1$ with $s_{h,1} + \cdots + s_{h,r_h} = s_h$ for each $h \in [k]$,*
- *integers $1 \leq \ell_{h,1} < \cdots < \ell_{h,r_h} \leq m$ for each $h \in [k]$,*
- *integers $t_{h,i} \geq s_{h,i}$ and bitstrings $b_{h,i} \in \{0,1\}^{t_{h,i}}$ and $q_{h,i} \in \binom{\{0,1\}^{t_{h,i}}}{s_{h,i}}$ for each $h \in [k]$ and $i \in [r_h]$*

with the property that there exist unique ordered restrictions $\beta_{1,1}, \dots, \beta_{h,r_h}$ such that, for all $h \in [k]$ and $i \in [r_h]$,

- **(i)** $(F_{h,\ell'}{\restriction}_{\gamma_{h,i}}){\restriction}_{\varrho_d} \equiv 0$ *for all* $1 \leq \ell' < \ell_{h,i}$,
- **(ii)** $(F_{h,\ell_{h,i}}{\restriction}_{\gamma_{h,i}}){\restriction}_{\varrho_d} \not\equiv 0$,
- **(iii)** $\mathcal{T}^{(b_{h,i})}_{\varrho_1,\dots,\varrho_{d-1}}(F_{h,\ell_{h,i}}{\restriction}_{\gamma_{h,i}}) = \beta_{h,i}$,
- **(iv)** $\beta_{h,i}$ *is ϱ_d-consistent and* $(F_{h,\ell_{h,i}}{\restriction}_{\gamma_{h,i}}){\restriction}_{\varrho_{d-1}\cup\beta_{h,i}} \equiv 1$,
- **(v)** *"$q_{h,i}$ identifies* $\mathrm{Stars}(\varrho_d)$ *within* $\mathrm{Dom}(\beta_{h,i}) \cap \mathrm{Stars}(\varrho_{d-1})$*" in the following sense: for all $j \in [t_{h,i}]$, we have $q_{h,i,j} = 1 \iff \mathrm{Stars}(\varrho_d)$ contains the j^{th} variable of $\mathrm{Dom}(\beta_{h,i}) \cap \mathrm{Stars}(\varrho_{d-1})$ in the order given by $\beta_{h,i}$.*

where

$$\gamma_{h,i} := (\beta_{1,1} \circ \cdots \circ \beta_{h,i-1}) \leftarrow_{\mathrm{Stars}(\varrho_{d-1})} d_{h,i},$$
$$d_{h,i} := (\, b_{1,1} \circ \cdots \circ b_{h,i-1}\,) \leftarrow^{q_{1,1} \circ \cdots \circ q_{h,i-1}} (c_h \circ \langle a_{h,1}, \ldots, a_{h,s_{h,1}+\cdots+s_{h,i-1}} \rangle).$$

(Note that conditions (iii) and (v) imply that $|\mathrm{Dom}(\beta_{h,i}) \cap \mathrm{Stars}(\varrho_{d-1})| = t_{h,i}$ and $|\mathrm{Dom}(\beta_{h,i}) \cap \mathrm{Stars}(\varrho_d)| = s_{h,i}$ and $\gamma_{h,i} = (\beta_{1,1} \circ \cdots \circ \beta_{h,i-1}) \leftarrow_{\mathrm{Stars}(\varrho_d)} (c_h \circ \langle a_{h,1}, \ldots, a_{h,s_{h,1}+\cdots+s_{h,i-1}} \rangle)$.)

Proof. Straightforward from Definition 19. ◀

▶ **Proposition 27** ("Serial depth-d switching lemma"). *Let $k, m \geq 1$, let $F_1, \ldots, F_k$ be depth-d formulas where $F_h = \mathrm{OR}(F_{h,1}, \ldots, F_{h,m})$ for each $h \in [k]$. Suppose $\boldsymbol{\varrho}_1 \subseteq \cdots \subseteq \boldsymbol{\varrho}_{d-1}$ are random restrictions such that, for all $r_1, \ldots, r_k \geq 1$ and $1 \leq \ell_{h,1} < \cdots < \ell_{h,r_h} \leq m$ and $t_{h,i} \geq 1$ and $b_{h,i} \in \{0,1\}^{t_{h,i}}$ and $d_{h,i} \in \{0,1\}^{t_{1,1}+\cdots+t_{h,i-1}}$ ($h \in [k]$ and $i \in [r_h]$),*

$$\mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_{d-1}} \Big[\, \exists \beta_{1,1}, \ldots, \beta_{k,r_k} \bigwedge_{\substack{h \in [k] \\ i \in [r_h]}} \Big(\mathcal{T}^{(b_{h,i})}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_{d-1}} (F_{h,\ell_{h,i}} \restriction_{(\beta_{1,1} \circ \cdots \circ \beta_{h,i-1}) \leftarrow_{\mathrm{Stars}(\varrho_{d-1})} d_{h,i}}) = \beta_{h,i} \Big) \,\Big] \leq \Big(\frac{1}{2e}\Big)^{t_{1,1}+\cdots+t_{k,r_k}}.$$

Then for all integers $s_1, \ldots, s_k \geq 1$ and bitstrings $a_h \in \{0,1\}^{s_h}$ and $c_h \in \{0,1\}^{s_1+\cdots+s_{h-1}}$ ($h \in [k]$) and $p \in [0,1]$, letting $\boldsymbol{\varrho}_d$ be a p-random refinement of $\boldsymbol{\varrho}_{d-1}$, we have

$$\mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_d} \Big[\, \exists \alpha_1, \ldots, \alpha_k \bigwedge_{h \in [k]} \Big(\mathcal{T}^{(a_h)}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_d} (F_h \restriction_{(\alpha_1 \circ \cdots \circ \alpha_{h-1}) \leftarrow_{\mathrm{Stars}(\boldsymbol{\varrho}_d)} c_h}) = \alpha_h \Big) \,\Big] \leq \Big(4ep(\ln m + 1)\Big)^{s_1+\cdots+s_k}.$$

Note that Proposition 25 is precisely the special case $k = 1$ of Proposition 27. The following proof closely parallels the proof of Proposition 25 (with a few more indices to keep track of). We omit the justifications of certain (in)equalities that would be redundant.

Proof. Fix $s_1, \ldots, s_k$ and $a_1, \ldots, a_k$ and $c_1, \ldots, c_k$ and p, and let $s := s_1 + \cdots + s_k$.

Let $\vec{r} = \{r_h\}$ and $\vec{s} = \{s_{h,i}\}$ and $\vec{\ell} = \{\ell_{h,i}\}$ and $\vec{t} = \{t_{h,i}\}$ and $\vec{b} = \{b_{h,i}\}$ and $\vec{q} = \{q_{h,i}\}$ (where $h \in [k]$ and $[i] \in [r_h]$) range over data satisfying the bullet items of Lemma 26. We have

$$\mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_d} \Big[\, \exists \alpha_1, \ldots, \alpha_k \bigwedge_{h \in [k]} \Big(\mathcal{T}^{(a_h)}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_d} (F_h \restriction_{(\alpha_1 \circ \cdots \circ \alpha_{h-1}) \leftarrow_{\mathrm{Stars}(\boldsymbol{\varrho}_d)} c_h}) = \alpha_h \Big) \,\Big]$$

$$\leq \sum_{\vec{r},\vec{s},\vec{\ell},\vec{t},\vec{b},\vec{q}} \mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_d} \left[\begin{array}{l} \exists \beta_{1,1}, \ldots, \beta_{k,r_k} \text{ such that, for all } h \in [k] \text{ and } i \in [r_h], \\ \quad \text{(i)} \;\; (F_{h,\ell'} \restriction_{\gamma_{h,i}}) \restriction_{\boldsymbol{\varrho}_d} \equiv 0 \text{ for all } 1 \leq \ell' < \ell_{h,i} \\ \quad \text{(ii)} \;\; (F_{h,\ell_{h,i}} \restriction_{\gamma_{h,i}}) \restriction_{\boldsymbol{\varrho}_d} \not\equiv 0 \\ \quad \text{(iii)} \;\; \mathcal{T}^{(b_{h,i})}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_{d-1}} (F_{h,\ell_{h,i}} \restriction_{\gamma_{h,i}}) = \beta_{h,i} \\ \quad \text{(iv)} \;\; \beta_{h,i} \text{ is } \boldsymbol{\varrho}_d\text{-consistent and } (F_{h,\ell_{h,i}} \restriction_{\gamma_{h,i}}) \restriction_{\boldsymbol{\varrho}_{d-1} \cup \beta_{h,i}} \equiv 1 \\ \quad \text{(v)} \;\; q_{h,i} \text{ identifies } \mathrm{Stars}(\boldsymbol{\varrho}_d) \text{ within } \mathrm{Dom}(\beta_{h,i}) \cap \mathrm{Stars}(\boldsymbol{\varrho}_{d-1}) \\ \text{where } \gamma_{h,i} := (\beta_{1,1} \circ \cdots \circ \beta_{h,i-1}) \leftarrow_{\mathrm{Stars}(\boldsymbol{\varrho}_{d-1})} d_{h,i}, \\ \qquad\quad d_{h,i} := (\, b_{1,1} \circ \cdots \circ b_{h,i-1}\,) \leftarrow^{q_{1,1} \circ \cdots \circ q_{h,i-1}} \\ \qquad\qquad\qquad\qquad (c_h \circ \langle a_{h,1}, \ldots, a_{h,s_{h,1}+\cdots+s_{h,i-1}} \rangle) \end{array} \right]$$

$$\leq 2^s \max_{\vec{r},\vec{s}} \sum_{\vec{\ell},\vec{t},\vec{b},\vec{q}} \mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_d} \Big[\, \exists \beta_{1,1}, \ldots, \beta_{k,r_k} \text{ such that (i)–(v) for all } h \in [k] \text{ and } i \in [r_h] \,\Big].$$

Letting $\boldsymbol{x}$ be a uniform random completion of $\boldsymbol{\varrho}_d$, we have

$$\begin{aligned}
&\mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\dots,\boldsymbol{\varrho}_d}\Big[\ \exists \beta_{1,1},\dots,\beta_{k,r_k} \text{ such that (i)–(v) for all } h\in[k] \text{ and } i\in[r_h]\ \Big]\\
&= 2^s \mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\dots,\boldsymbol{\varrho}_d,\boldsymbol{x}}\Big[\ \exists \beta_{1,1},\dots,\beta_{k,r_k} \text{ such that (i)–(v) and } \beta_{h,i}\subseteq\boldsymbol{x} \text{ for all } h\in[k],\, i\in[r_h]\ \Big]\\
&\le 2^s \mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\dots,\boldsymbol{\varrho}_d,\boldsymbol{x}}\left[\begin{array}{l}
\exists \beta_{1,1},\dots,\beta_{k,r_k} \text{ such that, for all } h\in[k] \text{ and } i\in[r_h],\\
\quad \text{(i')}\ \ (F_{h,\ell'}\restriction_{\gamma_{h,i}})(\boldsymbol{x})=0 \text{ for all } 1\le\ell'<\ell_{h,i}\\
\quad \text{(ii')}\ \ (F_{h,\ell_{h,i}}\restriction_{\gamma_{h,i}})\restriction(\boldsymbol{x})=1\\
\quad \text{(iii')}\ \ \mathcal{T}^{(b_{h,i})}_{\boldsymbol{\varrho}_1,\dots,\boldsymbol{\varrho}_{d-1}}(F_{h,\ell_{h,i}}\restriction_{\gamma_{h,i}})=\beta_{h,i}\\
\quad \text{(iv')}\ \ \beta_{h,i}\subseteq\boldsymbol{x}\\
\quad \text{(v')}\ \ q_{h,i} \text{ identifies } \mathrm{Stars}(\boldsymbol{\varrho}_d) \text{ within } \mathrm{Dom}(\beta_{h,i})\cap\mathrm{Stars}(\boldsymbol{\varrho}_{d-1})\\
\text{where } \gamma_{h,i}:=(\beta_{1,1}\circ\cdots\circ\beta_{h,i-1})\leftarrow_{\mathrm{Stars}(\boldsymbol{\varrho}_{d-1})} d_{h,i},\\
\qquad d_{h,i}:=(\,b_{1,1}\circ\cdots\circ b_{h,i-1}\,)\leftarrow^{q_{1,1}\circ\cdots\circ q_{h,i-1}}\\
\qquad\qquad (c_h\circ\langle a_{h,1},\dots,a_{h,s_{h,1}+\cdots+s_{h,i-1}}\rangle)
\end{array}\right]\\
&= (2p)^s(1-p)^{(t_{1,1}+\cdots+t_{k,r}-s)}\mu(\vec{\ell},\vec{t},\vec{b},\vec{q})\\
&\le (2p)^s\mu(\vec{\ell},\vec{t},\vec{b},\vec{q})
\end{aligned}$$

where

$$\mu(\vec{\ell},\vec{t},\vec{b},\vec{q}):=\mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\dots,\boldsymbol{\varrho}_d,\boldsymbol{x}}\Big[\ \exists \beta_{1,1},\dots,\beta_{k,r_k} \text{ such that (i')–(iv') for all } h\in[k] \text{ and } i\in[r_h]\ \Big].$$

Ignoring conditions (i') and (ii'), we have

$$\begin{aligned}
&\mu(\vec{\ell},\vec{t},\vec{b},\vec{q})\\
&\le \mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\dots,\boldsymbol{\varrho}_d,\boldsymbol{x}}\Big[\ \exists \beta_{1,1},\dots,\beta_{k,r_k} \text{ such that (iii') and (iv') for all } h\in[k] \text{ and } i\in[r_h]\ \Big]\\
&= \left(\frac{1}{2}\right)^{t_{1,1}+\cdots+t_{k,r_k}} \mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\dots,\boldsymbol{\varrho}_{d-1}}\Big[\ \exists \beta_{1,1},\dots,\beta_{k,r_k} \text{ such that (iii') for all } h\in[k] \text{ and } i\in[r_h]\ \Big]\\
&= \left(\frac{1}{2}\right)^{t_{1,1}+\cdots+t_{k,r_k}} \mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\dots,\boldsymbol{\varrho}_{d-1}}\left[\begin{array}{l}
\exists \beta_{1,1},\dots,\beta_{k,r_k} \text{ such that, for all } h\in[k] \text{ and } i\in[r_h],\\
\mathcal{T}^{(b_{h,i})}_{\boldsymbol{\varrho}_1,\dots,\boldsymbol{\varrho}_{d-1}}(F_{h,\ell_{h,i}}\restriction_{(\beta_{1,1}\circ\cdots\circ\beta_{h,i-1})\leftarrow_{\mathrm{Stars}(\boldsymbol{\varrho}_{d-1})}d_{h,i}})=\beta_{h,i} \text{ where}\\
d_{h,i}:=(\,b_{1,1}\circ\cdots\circ b_{h,i-1}\,)\leftarrow^{q_{1,1}\circ\cdots\circ q_{h,i-1}}\\
\qquad\qquad (c_h\circ\langle a_{h,1},\dots,a_{h,s_{h,1}+\cdots+s_{h,i-1}}\rangle)
\end{array}\right]\\
&\le \left(\frac{1}{4e}\right)^{t_{1,1}+\cdots+t_{k,r_k}}.
\end{aligned}$$

For each $h\in[k]$ and $i\in[r_h]$, let $q^\star_{h,i}$ range over functions associating each sequence $(\ell_{1,1},t_{1,1},b_{1,1},\dots,\ell_{h,i},t_{h,i},b_{h,i})$ with an element $q^\star_{h,i}(\ell_{1,1},t_{1,1},b_{1,1},\dots,\ell_{h,i},t_{h,i},b_{h,i})\in\{0,1\}^{t_{h,i}}_{s_{h,i}}$. Let

$$\mu(\vec{\ell},\vec{t},\vec{b},\vec{q}^\star):=\mu\Big(\vec{\ell},\vec{t},\vec{b},\langle q^\star_{1,1}(\ell_{1,1},t_{1,1},b_{1,1}),\dots,q^\star_{k,r_k}(\ell_{1,1},t_{1,1},b_{1,1},\dots,\ell_{k,r_k},t_{k,r_k},b_{k,r_k})\rangle\Big).$$

For every choice of $q^\star_{1,1},\dots,q^\star_{k,r_k}$, we have $\sum_{\vec\ell,\vec t,\vec b}\mu(\vec\ell,\vec t,\vec b,\vec{q^\star})\le 1$. Therefore, using Lemma 9,

$$\begin{aligned}\sum_{\vec\ell,\vec t,\vec b,\vec q}\mu(\vec\ell,\vec t,\vec b,\vec q) &\le \sum_{\ell_{1,1},t_{1,1},b_{1,1}}\binom{t_{1,1}}{s_{1,1}}\max_{q_{1,1}}\cdots\sum_{\ell_{k,r_k},t_{k,r_k},b_{k,r_k}}\binom{t_{k,r_k}}{s_{k,r_k}}\max_{q_{k,r_k}}\mu(\vec\ell,\vec t,\vec b,\vec q)\\ &= \max_{\vec{q^\star}}\sum_{\ell_{1,1},t_{1,1},b_{1,1}}\binom{t_{1,1}}{s_{1,1}}\cdots\sum_{\ell_{k,r_k},t_{k,r_k},b_{k,r_k}}\binom{t_{k,r_k}}{s_{k,r_k}}\mu(\vec\ell,\vec t,\vec b,\vec{q^\star})\\ &\le \max_{\vec{q^\star}}\sum_{\vec\ell,\vec t,\vec b}\binom{t_{1,1}+\cdots+t_{k,r_k}}{s}\mu(\vec\ell,\vec t,\vec b,\vec{q^\star})\\ &\le \max_{\vec{q^\star}} e^s\left(\frac1s\ln\left(\sum_{\vec\ell,\vec t,\vec b}e^{(t_{1,1}+\cdots+t_{k,r_k})}\mu(\vec\ell,\vec t,\vec b,\vec{q^\star})\right)+1\right)^s.\end{aligned}$$

Finally, we have

$$\begin{aligned}\sum_{\vec\ell,\vec t,\vec b}e^{(t_{1,1}+\cdots+t_{k,r_k})}\mu(\vec\ell,\vec t,\vec b,\vec{q^\star}) &\le \sum_{\vec\ell,\vec t,\vec b}\left(\frac14\right)^{t_{1,1}+\cdots+t_{k,r_k}}\\ &\le \sum_{\vec\ell}\sum_{t=r_1+\cdots+r_k}^{\infty}\left(\frac14\right)^t\sum_{\vec t,\vec b\colon t_{1,1}+\cdots+t_{k,r_k}=t}1\\ &= \sum_{\vec\ell}\sum_{t=r_1+\cdots+r_k}^{\infty}\left(\frac12\right)^t\binom{t-1}{r_1+\cdots+r_k-1}\\ &= \sum_{\vec\ell}1=\binom{m}{r_1}\cdots\binom{m}{r_k}\le m^{(r_1+\cdots+r_k)}\le m^s.\end{aligned}$$

Putting together these inequalities, we get the desired bound

$$\mathop{\mathbb{P}}_{\boldsymbol\varrho_1,\dots,\boldsymbol\varrho_d}\left[\ \exists\alpha_1,\dots,\alpha_k\bigwedge_{h\in[k]}\left(\mathcal{T}^{(a_h)}_{\boldsymbol\varrho_1,\dots,\boldsymbol\varrho_d}(F_h\restriction_{(\alpha_1\circ\cdots\circ\alpha_{h-1})\leftarrow\mathrm{Stars}(\boldsymbol\varrho_d)c_h})=\alpha_h\right)\ \right]\le\left(4ep(\ln m+1)\right)^s.\ \blacktriangleleft$$

8 Proof of Theorem 3

The following lemma is required for the analysis of depth-1 subformulas in the proof of Theorem 3.

▶ **Lemma 28.** *Let $\boldsymbol\varrho$ be a p-random restriction. Let $F_1,\dots,F_k$ be depth-1 formulas, let $s_1,\dots,s_k\ge1$, and let $a_h\in\{0,1\}^{s_h}$ and $c_h\in\{0,1\}^{s_1+\cdots+s_{h-1}}$ for each $h\in[k]$. Then*

$$\mathop{\mathbb{P}}_{\boldsymbol\varrho}\left[\ \exists\alpha_1,\dots,\alpha_k\bigwedge_{h\in[k]}\left(\mathcal{T}^{(a_h)}_{\boldsymbol\varrho}(F_h\restriction_{(\alpha_1\circ\cdots\circ\alpha_{h-1})\leftarrow\mathrm{Stars}(\boldsymbol\varrho)c_h})=\alpha_h\right)\ \right]\le(2p)^{s_1+\cdots+s_k}.$$

Proof. Assume $F_1\restriction_{\boldsymbol\varrho}$ computes a non-constant function, since otherwise $\mathcal{T}^{(a_1)}_{\boldsymbol\varrho}(F_1)$ does not exist (note that $\alpha_1=\langle\rangle$). Let V_1 be the set of variable indices occurring in literals of F_1. Observe that $\mathcal{T}^{(a_1)}_{\boldsymbol\varrho}(F_1)$ exists if, and only if, $|V_1\cap\mathrm{Stars}(\boldsymbol\varrho)|=s_1$ and $\boldsymbol\varrho$ gives the unique assignment to variables in $V_1\cap\mathrm{Dom}(\boldsymbol\varrho)$ such that $F_1\restriction_{\boldsymbol\varrho}$ is non-constant. This happens with probability $\binom{|V_1|}{s_1}p^{s_1}(\frac{1-p}{2})^{|V_1|-s_1}$, which is at most $(2p)^{s_1}$. Also note that this event only depends $\boldsymbol\varrho|_{V_1}$ (i.e., the partial function $\boldsymbol\varrho$ restricted to $\mathrm{Dom}(\boldsymbol\varrho)\cap V_1$). If this event holds, then we have $\mathcal{T}^{(a_1)}_{\boldsymbol\varrho}(F_1)=\alpha_1$ for the unique ordered restriction α_1 with $\mathrm{Dom}(\alpha_1)=V_1\cap\mathrm{Stars}(\boldsymbol\varrho)$ whose values are given by a_1.

Next let $F_2' := F_2\restriction_{\alpha_1 \leftarrow_{\mathrm{Stars}(\boldsymbol{\varrho})} c_1}$ and assume that $F_2'\restriction_{\boldsymbol{\varrho}}$ computes a non-constant function, since otherwise $\mathcal{T}_{\boldsymbol{\varrho}}^{(a_2)}(F_2')$ does not exist. Let V_2 be the set of variable indices occurring in literals of F_2', and note that $V_1 \cap V_2 = \emptyset$. Observe that $\mathcal{T}_{\boldsymbol{\varrho}}^{(a_2)}(F_2')$ exists if, and only if, $|V_2 \cap \mathrm{Stars}(\boldsymbol{\varrho})| = s_2$ and $\boldsymbol{\varrho}$ gives the unique assignment to variables in $V_2 \cap \mathrm{Dom}(\boldsymbol{\varrho})$ such that $F_2'\restriction_{\boldsymbol{\varrho}}$ is non-constant. Conditioned on the value of $\boldsymbol{\varrho}|_{V_1}$, this second event happens with probability $\binom{|V_2|}{s_2} p^{s_2} (\frac{1-p}{2})^{|V_2|-s_2}$, which is at most $(2p)^{s_2}$. Moreover, this second event only depends on $\boldsymbol{\varrho}|_{V_2}$.

Continuing in this manner, we conclude that the event in question holds with probability at most $(2p)^{s_1+\cdots+s_k}$. ◀

Proof of Theorem 3. Let F be a regular formula of depth $d+1$ and size s. For $i \in \{2,\ldots,d+1\}$, let m_i be the top fan-in of depth-i subformulas of F. Let

$$\lambda := \prod_{i=2}^{d+1} 8e^2(\ln m_i + 1).$$

Note that $s = m_2 \cdots m_{d+1}$ and therefore by Lemma 11

$$\lambda \leq (8e^2)^d \left(\frac{1}{d}\ln s + 1\right)^d.$$

We claim that F is λ-critical. To see this, consider any $p \in [0, \frac{1}{\lambda}]$. Let $\boldsymbol{\varrho}_1 \subseteq \cdots \subseteq \boldsymbol{\varrho}_{d+1}$ be a sequence of random restrictions where

- $\boldsymbol{\varrho}_1$ is a $\dfrac{1}{4e}$-random restriction,
- $\boldsymbol{\varrho}_i$ is a $\dfrac{1}{8e^2(\ln m_i + 1)}$-random refinement of $\boldsymbol{\varrho}_i$ for each $i \in \{2,\ldots,d\}$,
- $\boldsymbol{\varrho}_{d+1}$ is a $\dfrac{p\lambda}{16e(\ln m_{d+1}+1)}$-random refinement of $\boldsymbol{\varrho}_d$.

Note that $\boldsymbol{\varrho}_{d+1}$ is a p-random restriction.

For all integers $k \geq 1$ and $s_1,\ldots,s_k \geq 1$ and bitstrings $a_h \in \{0,1\}^{s_h}$ and $c_h \in \{0,1\}^{s_1+\cdots+s_{h-1}}$ ($h \in [k]$), we have:

- for all depth-1 subformulas $F_1,\ldots,F_k$ of F, Lemma 28 implies

$$\mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1}\left[\, \exists \alpha_1,\ldots,\alpha_k \bigwedge_{h\in[k]} \left(\mathcal{T}_{\boldsymbol{\varrho}_1}^{(a_h)}(F_h\restriction_{(\alpha_1\circ\cdots\circ\alpha_{h-1})\leftarrow_{\mathrm{Stars}(\boldsymbol{\varrho}_1)}c_h}) = \alpha_h\right) \,\right] \leq \left(\frac{1}{2e}\right)^{s_1+\cdots+s_k},$$

- for all $i \in \{2,\ldots,d\}$ and depth-i subformulas $F_1,\ldots,F_k$ of F, Proposition 27 implies

$$\mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_i}\left[\, \exists \alpha_1,\ldots,\alpha_k \bigwedge_{h\in[k]} \left(\mathcal{T}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_i}^{(a_h)}(F_h\restriction_{(\alpha_1\circ\cdots\circ\alpha_{h-1})\leftarrow_{\mathrm{Stars}(\boldsymbol{\varrho}_i)}c_h}) = \alpha_h\right) \,\right] \leq \left(\frac{1}{2e}\right)^{s_1+\cdots+s_k},$$

- finally, Proposition 25 (or Proposition 27) implies

$$\mathop{\mathbb{P}}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_{d+1}}\left[\, \mathcal{T}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_{d+1}}^{(a_1)}(F) \text{ exists } \,\right] \leq \left(\frac{p\lambda}{4}\right)^{s_1}.$$

Therefore, for all $t \geq 1$, we have

$$\begin{aligned}\mathbb{P}\left[\, \mathsf{DT}_{\mathsf{depth}}(F\restriction \boldsymbol{R}_p) \geq t \,\right] &\leq \sum_{u=t}^{\infty} \mathbb{P}\left[\, \mathcal{T}^{*}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_{d+1}}(F) \text{ has depth } u \,\right] \\ &\leq \sum_{u=t}^{\infty} \sum_{a\in\{0,1\}^u} \mathbb{P}\left[\, \mathcal{T}^{(a)}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_{d+1}}(F) \text{ exists} \,\right] \\ &\leq \sum_{u=t}^{\infty} 2^u \left(\frac{p\lambda}{4}\right)^u \leq (p\lambda)^t.\end{aligned}$$

This shows that F is λ-critical, and since $\lambda = O(\frac{1}{d}\log s)^d$, the theorem is proved. ◀

9 Satisfiability Algorithms

The following theorem gives a randomized #SAT algorithm for regular AC^0 formulas. For AC^0 circuits of size $\Omega(n^2)$ (after converting to regular formulas), this matches the runtime of the #SAT algorithm of Impagliazzo, Matthews and Paturi [7].

▶ **Theorem 29.** *There is a randomized, zero-error algorithm which, given a regular* AC^0 *formula* F *of depth* $d+1$ *and size* s *on* n *variables, outputs a decision tree for* F *of size* $O(sn\cdot 2^{(1-\varepsilon)n})$ *where* $\varepsilon = 1/O(\frac{1}{d}\log s)^d$. *This algorithm also solves the* #SAT *problem, that is, it counts the number of satisfying assignments for* F.

Proof. First we require a lemma that $|\mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F\restriction_\gamma)| \leq |\mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F)|$ for any depth-d formulas F, restrictions $\varrho_1 \subseteq \cdots \subseteq \varrho_d$ and ϱ_d-consistent restriction γ. This is straightforward from Definition 19 of $\mathcal{T}_{\varrho_1,\ldots,\varrho_d}(\cdot)$, but requires a careful argument (by induction on a stronger statement) to make precise. We omit the details.

Accepting this claim, we can extract from Definition 19 an algorithm which, given any depth-d formula F and restrictions $\varrho_1 \subseteq \cdots \subseteq \varrho_d$, computes the set $\mathcal{T}_{\varrho_1,\ldots,\varrho_d}(F)$ in time $O(n)\cdot\sum_{i=1}^{d}\sum_{F_i}|\mathcal{T}_{\varrho_1,\ldots,\varrho_i}(F_i)|$ where F_i ranges over depth-i subformulas of F.

If we are now given a regular AC^0 formula F of depth $d+1$ and size s on n variables, we can compute a decision tree for F as follows. Consider any sets $D_1 \subseteq \cdots \subseteq D_{d+1} \subseteq [n]$ and let $D := D_{d+1}$. We get a decision tree for F by querying all variables in D, receiving answers $\varrho : D \to \{0,1\}$, and then proceeding as the decision tree $\mathcal{T}_{\varrho_1,\ldots,\varrho_{d+1}}(F)$ where ϱ_i is the restriction of ϱ to domain D_i. The time required to construct this decision tree is $O(n)\cdot\sum_{i=1}^{d+1}\sum_{F_i}\sum_{\varrho:D\to\{0,1\}}|\mathcal{T}_{\varrho_1,\ldots,\varrho_i}(F_i)|$. If $\boldsymbol{\varrho}$ is a uniform random restriction with domain D and $|D| \leq (1-\varepsilon)n$ (for a choice of $\delta > 0$ to be determined), then this bound is $O(n\cdot 2^{(1-\delta)n})\cdot\sum_{i=1}^{d+1}\sum_{F_i}\mathbb{E}\,|\mathcal{T}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_i}(F_i)|$.

Let us now randomly generate sets $\boldsymbol{D}_1 \subseteq \cdots \subseteq \boldsymbol{D}_{d+1} \subseteq [n]$ as follows:

- $\boldsymbol{D}_1$ is a $1-\frac{1}{4e}$-random subset of $[n]$ (i.e., $\boldsymbol{D}_1$ includes each variable index in $[n]$ independently with probability $1-\frac{1}{4e}$),
- for each $i \in \{2,\ldots,d+1\}$, $\boldsymbol{D}_i$ is the union of $\boldsymbol{D}_{i-1}$ and a $1-\frac{1}{8e^2(\ln m_i+1)}$-random subset of $[n]\setminus\boldsymbol{D}_{i-1}$ where m_i is the top fan-in of depth-i subformulas of F (equivalently: $[n]\setminus\boldsymbol{D}_i$ is a $\frac{1}{8e^2(\ln m_i+1)}$-random subset of $[n]\setminus\boldsymbol{D}_{i-1}$).

Note that $\boldsymbol{D}$ $(:= \boldsymbol{D}_{d+1})$ is a $(1-\frac{1}{\lambda})$-random subset of $[n]$ where $\lambda = O(\frac{1}{d}\log s)^d$. The proof of Theorem 3 shows that, for every $i \in [d+1]$ and depth-i subformula F_i,

$$\mathbb{E}\,|\mathcal{T}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_i}(F_i)| = \sum_{t=0}^{\infty}\sum_{a\in\{0,1\}^t}\mathbb{P}\left[\, \mathcal{T}^{(a)}_{\boldsymbol{\varrho}_1,\ldots,\boldsymbol{\varrho}_i}(F_i) \text{ exists} \,\right] \leq 1 + \sum_{t=1}^{\infty} 2^t(1/2e)^t = \frac{1}{1-(1/e)}.$$

We may assume that $\lambda \leq n/2$, since otherwise the theorem is trivial. We then have $\mathbb{P}[\ |\boldsymbol{D}| \leq (1-\frac{1}{2\lambda})n\ ]$ with probability $\Omega(1)$ (and in fact $1-o(1)$ when $\lambda \ll n$). It follows that, with at least this probability, our algorithm constructs a decision tree for F in time $O(sn \cdot 2^{(1-\frac{1}{2\lambda})n})$. ◀

As a corollary, we improve the parameters of an algorithm of Santhanam and Williams [14] for the satisfiability problem for q-QB-CNF and q-QB-DNF, the class of quantified CNF and DNF formulas with q quantifier blocks (i.e., q quantifier alternations).

▶ **Corollary 30.** *Satisfiability of* q-QB-CNF *(resp.* q-QB-DNF*) with* n *variables and* $\mathrm{poly}(n)$ *clauses (resp. disjuncts) can be solved probabilistically with zero error in time* $\mathrm{poly}(n) \cdot 2^{n-\Omega(qn^{1/(q+1)})+O(q)}$.

The proof of Corollary 30 is adapted straightforwardly from [14], using Theorem 29 in place of the AC^0-circuit satisfiability algorithm of Impagliazzo, Matthews and Paturi [7]. Corollary 30 extends from $o(\frac{\log n}{\log\log n})$ to $o(\log n)$ the range of q for which the algorithm of [14] beats exhaustive search (i.e., $\mathrm{poly}(n) \cdot 2^n$ time). We remark that a second algorithm in [14] running time $\mathrm{poly}(n) \cdot 2^{n-\Omega(q)}$, which beats exhaustive search when $q = \omega(\log n)$. The range of q where q-QB-CNF and q-QB-DNF is not known to beat exhaustive search by a factor of at least n^k is therefore reduced to between $c_1 \log n$ and $c_2 \log n$ for constants $c_1(k) < c_2(k)$.

10 Open Questions

It is an open question whether the assumption of regularity is unnecessary in Theorem 3. We conjecture that our criticality bound for regular formulas holds for all formula.

▶ **Conjecture 1.** *All* AC^0 *formulas of depth* $d+1$ *and size* s *have criticality at most* $O(\frac{1}{d}\log s)^d$.

For (regular) formulas of n variables, can this bound be improved to $O(\frac{1}{d}\log(\frac{s}{n})+\log(d))^d$? (Results in [7] for AC^0 circuits involve the quantity $O(\log(\frac{s}{n})+d\log(d))^d$.)

Since $\mathsf{deg}(f) \leq \mathsf{DT}_{\mathsf{depth}}(f)$ for all boolean functions f, it follows that λ-criticality implies λ-*degree-criticality*, that is, the bound $\mathbb{P}[\ \mathsf{deg}(f\restriction_{\boldsymbol{R}_p}) \geq t\ ] \leq (p\lambda)^t$. What about a reserve implication?

▶ Question 1. Does degree-criticality λ imply criticality $O(\lambda)$?

Tal [16] showed that DeMorgan formulas of size L have degree-criticality $O(\sqrt{L})$. As a special case of Question 1, one can ask:

▶ Question 2. Do DeMorgan formulas of size L have criticality $O(\sqrt{L})$?

Finally, we ask a question that would potentially yield a much simpler and more aesthetic proof of Theorem 3. We will say that a boolean function f is *hereditarily* λ*-critical* if every subfunction of f is λ-critical (i.e., $f\restriction_\varrho$ is λ-critical for every restriction ϱ).

▶ Question 3. Suppose f is the disjunction of boolean functions $f_1 \vee \cdots \vee f_m$ where each f_i is hereditarily λ-critical. Is f necessarily $O(\lambda \ln(m+1))$ critical?

A positive answer to Question 3 implies Theorem 3. If Question 3 could be answered affirmatively, we may then consider the following generalization:

▶ Question 4. Suppose f is the disjunction of boolean functions $f_1 \vee \cdots \vee f_m$ where each f_i is hereditarily λ_i-critical. Let $\lambda \geq \max\{\lambda_1, \ldots, \lambda_m\}$ such that $\sum_{i=1}^m e^{-(\lambda/\lambda_i)} \leq 1$. Is f necessarily $O(\lambda)$ critical?

A positive answer to Question 4 would be very interesting as it implies Conjecture 1.

References

1 Kazuyuki Amano. Tight Bounds on the Average Sensitivity of k-CNF. *Theory of Computing*, 7(1):45–48, 2011.

2 Paul Beame. A switching lemma primer. Technical report, Technical Report UW-CSE-95-07-01, Department of Computer Science and Engineering, University of Washington, 1994.

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 *27th Annual IEEE Conference on Computational Complexity*, pages 117–125, 2012.

4 Ravi B Boppana. The average sensitivity of bounded-depth circuits. *Information processing letters*, 63(5):257–261, 1997.

5 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. ACM, 1986.

6 Johan Håstad. On the correlation of parity and small-depth circuits. *SIAM Journal on Computing*, 43(5):1699–1708, 2014.

7 Russell Impagliazzo, William Matthews, and Ramamohan Paturi. A satisfiability algorithm for AC^0. In *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 961–972. SIAM, 2012.

8 Nathan Keller and Noam Lifshitz. Approximation of biased Boolean functions of small total influence by DNF's. *arXiv preprint*, 2017. `arXiv:1703.10116`.

9 Nathan Linial, Yishay Mansour, and Noam Nisan. Constant depth circuits, Fourier transform, and learnability. *Journal of the ACM*, 40(3):607–620, 1993.

10 Alexander A Razborov. An equivalence between second order bounded domain bounded arithmetic and first order bounded arithmetic, 1993.

11 Benjamin Rossman. An entropy proof of the switching lemma and tight bounds on the decision-tree size of AC^0 circuits.

12 Benjamin Rossman. The average sensitivity of bounded-depth formulas. *Computational Complexity*, 27(2):209–223, 2018.

13 Benjamin Rossman and Srikanth Srinivasan. Separation of $AC^0[\oplus]$ Formulas and Circuits. In *LIPIcs-Leibniz International Proceedings in Informatics*, volume 80. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

14 Rahul Santhanam and Ryan Williams. Beating exhaustive search for quantified boolean formulas and connections to circuit complexity. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 231–241. SIAM, 2014.

15 Dominik Scheder and Li-Yang Tan. On the average sensitivity and density of k-CNF formulas. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 683–698. Springer, 2013.

16 Avishay Tal. Shrinkage of De Morgan formulae by spectral techniques. In *55th Annual IEEE Symposium on Foundations of Computer Science*, pages 551–560, 2014.

17 Avishay Tal. Tight bounds on the Fourier spectrum of AC^0. In *LIPIcs-Leibniz International Proceedings in Informatics*, volume 79. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

18 Patrick Traxler. Variable Influences in Conjunctive Normal Forms. In *Theory and Applications of Satisfiability Testing-SAT 2009: 12th International Conference, SAT 2009, Swansea, UK, June 30-July 3, 2009. Proceedings*, volume 5584, page 101. Springer, 2009.

19 Andre M. Zubkov and Aleksandr A. Serov. A complete proof of universal inequalities for the distribution function of the binomial law. *Theory of Probability & Its Applications*, 57(3):539–544, 2013.

A Appendix: Average Sensitivity of Size-m DNF Formulas

As a warm-up for our switching lemma for size-m DNF formulas (Section 4), we present a simple proof that every size-m DNF formula F with expected value $\lambda \in [0,1]$ has average sensitivity at most $\min\{2\log(m+1), 2\lambda\log(m/\lambda)\}$. Up to an $1+o(1)$ factor, these bounds can be derived from known results on the average sensitivity of width-w DNFs (see Remark 33). However, our proof involves different argument based on the entropy of the "first witness function" associated with F. This argument was the starting point for our alternative proof of the switching lemma and provides a simple illustration of the underlying principle.

Recall the definitions of *sensitivity* and *average sensitivity*. For a function f with domain $\{0,1\}^n$ and a point $x \in \{0,1\}^n$, let

$$\mathsf{S}(f,x) := |\{i \in [n] : f(x) \neq f(x \oplus i)\}| \quad \text{and} \quad \mathsf{AS}(f) := \mathop{\mathbb{E}}_{x\in\{0,1\}^n}[\, \mathsf{S}(f,x)\,].$$

The *expected value* of f is $\mathbb{E}_{x\in\{0,1\}^n}[\, f(x)\,]$.

▶ **Theorem 31.** *Every m-clause DNF with expected value λ has average sensitivity at most* $\min\{2\log(m+1), 2\lambda\log(m/\lambda)\}$.

Proof. Let $F = C_1 \vee \cdots \vee C_m$ be an m-clause DNF. Let $\widetilde{F} : \{0,1\}^n \to [m+1]$ be the "first witness function" mapping $x \in \{0,1\}^n$ to the index of the first satisfied clause if any, and otherwise to $m+1$. Let

$$\mathsf{S}_<(\widetilde{F},x) := |\{i \in [n] : \widetilde{F}(x) < \widetilde{F}(x \oplus i)\}| \quad \text{and} \quad \mathsf{AS}_<(\widetilde{F}) := \mathop{\mathbb{E}}_{x\in\{0,1\}^n}[\, \mathsf{S}_<(\widetilde{F},x)\,].$$

Observe that $\mathsf{AS}(F) \le \mathsf{AS}(\widetilde{F}) = 2{\cdot}\mathsf{AS}_<(\widetilde{F})$.

Let $\mu = (\mu_1, \ldots, \mu_{m+1})$ be the probability distribution induced by $\widetilde{F}$ under the uniform distribution on $\{0,1\}^n$, that is, $\mu_\ell := \mathbb{P}_{x\in\{0,1\}^n}[\, \widetilde{F}(x) = \ell\,]$. For each $\ell \in [m]$, we have

$$\begin{aligned}
2^{\mathbb{E}_{y\in\widetilde{F}^{-1}(\ell)}[\,\mathsf{S}_<(\widetilde{F},y)\,]} &\le \mathop{\mathbb{E}}_{y\in\widetilde{F}^{-1}(\ell)}[\, 2^{\mathsf{S}_<(\widetilde{F},y)}\,] && \text{by Jensen's inequality}\\
&\le 2^{|C_\ell|} && \text{since } \mathsf{S}_<(\widetilde{F},y) \le |C_\ell| \text{ for all } y \in \widetilde{F}^{-1}(\ell)\\
&\le \frac{1}{\mu_\ell} && \text{since } \mu_\ell \le \mathop{\mathbb{P}}_{x\in\{0,1\}^n}[\, C_\ell(x) = 1\,] = 2^{-|C_\ell|}.
\end{aligned}$$

Therefore, $\mathbb{E}_{y\in\widetilde{F}^{-1}(\ell)}[\, \mathsf{S}_<(\widetilde{F},y)\,] \le \log(1/\mu_\ell)$.

Using the fact that μ has entropy at most $\log(m+1)$, we have

$$\begin{aligned}
\mathsf{AS}_<(\widetilde{F}) &= \mathop{\mathbb{E}}_{x\in\{0,1\}^n}[\, \mathsf{S}_<(\widetilde{F},x)\,]\\
&= \sum_{\ell\in[m]} \mu_\ell \mathop{\mathbb{E}}_{y\in\widetilde{F}^{-1}(\ell)}[\, \mathsf{S}_<(\widetilde{F},y)\,]\\
&\le \sum_{\ell\in[m]} \mu_\ell \log(1/\mu_\ell) \;\le\; \sum_{\ell\in[m+1]} \mu_\ell \log(1/\mu_\ell) \;=\; \mathbb{H}(\mu) \;\le\; \log(m+1).
\end{aligned}$$

We conclude that $\mathsf{AS}(F) \le 2\log(m+1)$.

If F has expected value λ, then letting $\mu'_\ell := \mu_\ell/\lambda$ (and noting that $\lambda = \sum_{\ell\in[m]} \mu_\ell$), we have

$$\sum_{\ell\in[m]} \mu_\ell \log(1/\mu_\ell) = \lambda \sum_{\ell\in[m]} \mu'_\ell\Big(\log(1/\mu'_\ell) - \log(\lambda)\Big) = \lambda\Big(\mathbb{H}(\mu') - \log(\lambda)\Big) \le \lambda\log(m/\lambda).$$

This gives the bound $\mathsf{AS}(F) \le 2\lambda\log(m/\lambda)$. ◀

For $k, t \in \mathbb{N}$, observe that the function $\mathrm{PARITY}(x_1, \ldots, x_k) \wedge \mathrm{AND}(x_{k+1}, \ldots, x_{k+t})$ is equivalent to a DNF with $m := 2^{k-1}$ clauses and has expected value $\lambda := (1/2)^{t+1}$ and average sensitivity $(k+t)2^{-t}$ $(= 2\lambda \log(m/\lambda))$. This shows that Theorem 31 is tight whenever $\lambda \in [0, \frac{1}{2}]$ is an inverse power of two.

▶ Remark 32. Theorem 31 has a (weak) converse: Keller and Lifshitz [8] showed that every boolean function with expected value λ and average sensitivity at most $2\lambda \log(m/\lambda)$ is $\varepsilon\lambda$-approximated by a DNF of size $2^{m^{O(1/\varepsilon)}}$.

▶ Remark 33. The average sensitivity of a width-w DNF with expected value λ is known to be at most the minimum of w (Amano [1]), $2\lambda w$ (Boppana [4]) and $2(1-\lambda)w/\log(\frac{1}{1-\lambda})$ (Traxler [18]). Each of these bounds is tight for a certain values of λ. Extending all three bounds, Scheder and Tan [15] proved an upper bound of $\beta(\lambda)w$ for a certain piecewise linear function $\beta : [0,1] \to [0,1]$; this bound is asymptotically tight for all values of λ. By approximating any m-clause by a DNF of width $\lceil \log m \rceil$, they also observe that $(1 + o(1))\beta(\lambda) \log(m+1)$ is an upper bound on the average sensitivity of m-clause DNFs.

Almost Optimal Distribution-Free Junta Testing

Nader H. Bshouty
Department of Computer Science, Technion, Haifa, Israel
bshouty@cs.technion.ac.il

Abstract

We consider the problem of testing whether an unknown n-variable Boolean function is a k-junta in the distribution-free property testing model, where the distance between functions is measured with respect to an arbitrary and unknown probability distribution over $\{0,1\}^n$. Chen, Liu, Servedio, Sheng and Xie [35] showed that the distribution-free k-junta testing can be performed, with one-sided error, by an adaptive algorithm that makes $\tilde{O}(k^2)/\epsilon$ queries. In this paper, we give a simple two-sided error adaptive algorithm that makes $\tilde{O}(k/\epsilon)$ queries.

2012 ACM Subject Classification Mathematics of computing; Mathematics of computing → Discrete mathematics; Mathematics of computing → Probabilistic algorithms; Theory of computation → Probabilistic computation

Keywords and phrases Distribution-free property testing, k-Junta

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.2

Related Version `https://arxiv.org/abs/1901.00717`

Acknowledgements We would like to thank Xi Chen for reading the early version of the paper and for verifying the correctness of the algorithm.

1 Inroduction

Property testing of Boolean function was first considered in the seminal works of Blum, Luby and Rubinfeld [11] and Rubinfeld and Sudan [42] and has recently become a very active research area. See for example, [1, 2, 3, 4, 7, 8, 13, 14, 15, 16, 18, 19, 22, 24, 27, 29, 32, 33, 37, 36, 39, 43] and other works referenced in the surveys [26, 40, 41].

A function $f : \{0,1\}^n \to \{0,1\}$ is said to be k-junta if it depends on at most k variables. Juntas have been of particular interest to the computational learning theory community [9, 10, 12, 30, 34, 38]. A problem closely related to learning juntas is the problem of testing juntas: Given black-box query access to a Boolean function f. Distinguish, with high probability, the case that f is k-junta versus the case that f is ϵ-far from every k-junta.

In the uniform distribution framework, where the distance between two functions is measured with respect to the uniform distribution, Ficher et al. [24] introduced the junta testing problem and gave adaptive and non-adaptive algorithms that make $poly(k)/\epsilon$ queries. Blais in [5] gave a non-adaptive algorithm that makes $\tilde{O}(k^{3/2})/\epsilon$ queries and in [6] an adaptive algorithm that makes $O(k \log k + k/\epsilon)$ queries. On the lower bounds side, Fisher et al. [24] gave an $\Omega(\sqrt{k})$ lower bound. Chockler and Gutfreund [21] gave an $\Omega(k)$ lower bound for adaptive testing and, recently, Sağlam in [43] improved this lower bound to $\Omega(k \log k)$. For the non-adaptive testing Chen et al. [17] gave the lower bound $\tilde{\Omega}(k^{3/2})/\epsilon$.

In the *distribution-free property testing*, [28], the distance between Boolean functions is measured with respect to an arbitrary and unknown distribution $\mathcal{D}$ over $\{0,1\}^n$. In this model, the testing algorithm is allowed (in addition to making black-box queries) to draw random $x \in \{0,1\}^n$ according to the distribution $\mathcal{D}$. This model is studied in [20, 23, 25, 31, 35]. For testing k-junta in this model, Chen et al. [35] gave a one-sided adaptive algorithm that makes $\tilde{O}(k^2)/\epsilon$ queries and proved a lower bound $\Omega(2^{k/3})$ for any non-adaptive algorithm.

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka; Article No. 2; pp. 2:1–2:13
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

The results of Halevy and Kushilevitz [31] gives a one-sided non-adaptive algorithm that makes $O(2^k/\epsilon)$ queries. The adaptive $\Omega(k \log k)$ uniform-distribution lower bound from [43] trivially extend to the distribution-free model.

In this paper, we close the gap between the adaptive lower and upper bound. We prove

▶ **Theorem 1.** *For any $\epsilon > 0$, there is a two-sided distribution-free adaptive algorithm for ϵ-testing k-junta that makes $\tilde{O}(k/\epsilon)$ queries.*

Our exact upper bound is $O((k/\epsilon)\log(k/\epsilon))$ and therefore, by Sağlam [43] lower bound of $\Omega(k \log k)$, our bound is tight for any constant ϵ.

2 Preliminaries

In this section we give some notations follows by a formal definition of the model and some preliminary known results

2.1 Notations

We start with some notations. Denote $[n] = \{1, 2, \ldots, n\}$. For $S \subseteq [n]$ and $x = (x_1, \ldots, x_n)$ we write $x(S) = \{x_i | i \in S\}$. For $X \subset [n]$ we denote by $\{0,1\}^X$ the set of all binary strings of length $|X|$ with coordinates indexed by $i \in X$. For $x \in \{0,1\}^n$ and $X \subseteq [n]$ we write $x_X \in \{0,1\}^X$ to denote the projection of x over coordinates in X. We denote by 1_X and 0_X the all one and all zero strings in $\{0,1\}^X$, respectively. When we write $x_I = 0$ we mean $x_I = 0_I$. For $X_1, X_2 \subseteq [n]$ where $X_1 \cap X_2 = \emptyset$ and $x \in \{0,1\}^{X_1}, y \in \{0,1\}^{X_2}$ we write $x \circ y$ to denote their concatenation, the string in $\{0,1\}^{X_1 \cup X_2}$ that agrees with x over coordinates in X_1 and agrees with y over X_2. For $X \subseteq [n]$ we denote $\overline{X} = [n] \backslash X$. We say that the Boolean function $f : \{0,1\}^n \to \{0,1\}$ is a literal if $f \in \{x_1, \ldots, x_n, \bar{x_1}, \ldots, \bar{x_n}\}$.

Given $f, g : \{0,1\}^n \to \{0,1\}$ and a probability distribution $\mathcal{D}$ over $\{0,1\}^n$, we say that f is *ϵ-close to g with respect to $\mathcal{D}$* if $\mathbf{Pr}_{x \in \mathcal{D}}[f(x) \neq g(x)] \leq \epsilon$, where $x \in \mathcal{D}$ means x is chosen from $\{0,1\}^n$ according to the distribution $\mathcal{D}$. We say that f is *ϵ-far from g with respect to $\mathcal{D}$* if $\mathbf{Pr}_{x \in \mathcal{D}}[f(x) \neq g(x)] \geq \epsilon$. We say that f is *ϵ-far from every k-junta with respect to $\mathcal{D}$* if for every k-junta g, f is ϵ-far from g with respect to $\mathcal{D}$. We will use U to denote the uniform distribution over $\{0,1\}^n$.

2.2 The Model

In this subsection, we define the model.

We consider the problem of testing juntas in the distribution-free testing model. In this model, the algorithm has access to a k-junta f via a black-box that returns $f(x)$ when a string x is queried, and access to unknown distribution $\mathcal{D}$ via an oracle that returns $x \in \{0,1\}^n$ chosen randomly according to the distribution $\mathcal{D}$.

A *distribution-free testing algorithm* $\mathcal{A}$ is a algorithm that, given as input a distance parameter ϵ and the above two oracles,

1. if f is k-junta then $\mathcal{A}$ output "accept" with probability at least $2/3$.
2. if f is ϵ-far from every k-junta with respect to the distribution $\mathcal{D}$ then it output "reject" with probability at least $2/3$.

We say that $\mathcal{A}$ is *one-sided* if it always accepts when f is k-junta, otherwise, it is called *two sided* algorithm. The *query complexity* of a distribution-free testing algorithm is the number of queries made on f.

2.3 Preliminaries Results

In this section, we give some known results that will be used in the sequel.

For a Boolean function f and $X \subset [n]$, we say that X is a *relevant set* of f if there are $a, b \in \{0,1\}^n$ such that $f(a) \neq f(b_X \circ a_{\overline{X}})$. When $X = \{i\}$ then we say that x_i is *relevant variable* of f. Obviously, if X is relevant set of f then $x(X)$ contains at least one relevant variable of f. In particular, we have

▶ **Lemma 2.** *If $\{X_i\}_{i\in[r]}$ is a partition of $[n]$ then for any Boolean function f the number of relevant sets X_i of f is at most the number of relevant variables of f.*

We will use the following folklore result that is formally proved in [35].

▶ **Lemma 3.** *Let $\{X_i\}_{i\in[r]}$ be a partition of $[n]$. Let f be a Boolean function and $u \in \{0,1\}^n$. If $f(u) \neq f(0)$ then a relevant set X_ℓ of f with a string $v \in \{0,1\}^n$ that satisfies $f(v) \neq f(0_{X_\ell} \circ v_{\overline{X_\ell}})$ can be found with $\lceil \log_2 r \rceil$ queries.*

The following is from [6]

▶ **Lemma 4.** *There exists a one-sided adaptive algorithm, **UniformJunta**(f, k, ϵ, δ), for ϵ-testing k-junta that makes $O(((k/\epsilon)+k\log k)\log(1/\delta))$ queries and rejects f with probability at least $1-\delta$ when it is ϵ-far from every k-junta with respect to the uniform distribution.*

The following is from [35].

▶ **Lemma 5.** *Let $\mathcal{D}$ be any probability distribution over $\{0,1\}^n$. If f is ϵ-far from every k-junta with respect to $\mathcal{D}$ then for any $J \subseteq [n]$, $|J| \leq k$ we have*

$$\boldsymbol{Pr}_{x\in\mathcal{D},y\in U}[f(x) \neq f(x_J \circ y_{\bar{J}})] \geq \epsilon.$$

Proof. Let $J \subseteq [n]$ of size $|J| \leq k$. For every fixed $y \in \{0,1\}^n$ the function $f(x_J \circ y_{\bar{J}})$ is k-junta and therefore $\mathbf{Pr}_{x\in\mathcal{D}}[f(x) \neq f(x_J \circ y_{\bar{J}})] \geq \epsilon$. Therefore

$$\mathbf{Pr}_{x\in\mathcal{D},y\in U}[f(x) \neq f(x_J \circ y_{\bar{J}})] \geq \epsilon.$$ ◀

3 The Algorithm

In this section, we prove the correctness of the algorithm and show that it makes $\tilde{O}(k/\epsilon)$ queries. We first give an overview of the algorithm then prove its correctness and analyze its query complexity.

3.1 Overview of the Algorithm

In this subsection we give an overview of the algorithm. We will use the notation in Subsection 2.1 and the definitions and Lemmas in Subsection 2.3.

Consider the algorithm in Figure 1. In steps 1-2, the algorithm uniformly at random partitions $[n]$ into $r = 2k^2$ disjoint sets $X_1, \ldots, X_r$. Lemma 6 shows that,

▶ **Fact 1.** *If the function is k-junta then with high probability (w.h.p), each set of variables $x(X_i) = \{x_j | j \in X_i\}$ contains at most one relevant variable.*

In steps 3-12, the algorithm finds

▶ **Fact 2.** *relevant sets $\{X_i\}_{i\in I}$ such that for $X = \cup_{i\in I} X_i$, w.h.p., the function $f(x_X \circ 0_{\overline{X}})$ is $\epsilon/2$-close to f with respect to $\mathcal{D}$.*

To find such set, the algorithm, after finding relevant sets $\{X_i\}_{i\in I'}$, chooses random string $u \in \mathcal{D}$ and tests if $f(u_{X'} \circ 0_{\overline{X'}}) \neq f(u)$ where $X' = \cup_{i\in I'} X_i$. The variable $t(X')$ counts for how many random strings $u \in \mathcal{D}$ we get $f(u_{X'} \circ 0_{\overline{X'}}) = f(u)$. If $t(X')$ reaches the value $O((\log k)/\epsilon)$ then, w.h.p, $f(x_{X'} \circ 0_{\overline{X'}})$ is $\epsilon/2$-close to f with respect to $\mathcal{D}$ and $X = X'$. Otherwise, $f(u_{X'} \circ 0_{\overline{X'}}) \neq f(u)$ and using Lemma 3 the algorithm finds a new relevant set X_ℓ. This is proved in Lemma 10.

In addition, for each relevant set X_ℓ, $\ell \in I$, it finds a string $v^{(\ell)}$ that satisfies $f(v^{(\ell)}) \neq f(0_{X_\ell} \circ v^{(\ell)}_{\overline{X_\ell}})$. Obviously, if $|I| > k$ then, since each relevant set contains at least one relevant variable, the target is not k-junta and the algorithm rejects. See Lemma 2.

Now one of the key ideas is the following: If f is k-junta then $f(x_X \circ 0_{\overline{X}})$ is k-junta. If f is ϵ-far from every k-junta with respect to $\mathcal{D}$ then since, by Fact 2, w.h.p., $f(x_X \circ 0_{\overline{X}})$ is $\epsilon/2$-close to f with respect to $\mathcal{D}$ we have that,

▶ **Fact 3.** *If f is ϵ-far from every k-junta with respect to $\mathcal{D}$ then, w.h.p., $f(x_X \circ 0_{\overline{X}})$ is $\epsilon/2$-far from every k-junta with respect to $\mathcal{D}$.*

Now, since each X_ℓ, $\ell \in I$ is relevant set and $f(v^{(\ell)}) \neq f(0_{X_\ell} \circ v^{(\ell)}_{\overline{X_\ell}})$, for $\ell \in I$ the function $f(x_{X_\ell} \circ v^{(\ell)}_{\overline{X_\ell}})$ is non-constant. In steps 13-17, the algorithm tests that,

▶ **Fact 4.** *w.h.p., for each $\ell \in I$ there is $\tau(\ell) \in X_\ell$ such that $f(x_{X_\ell} \circ v^{(\ell)}_{\overline{X_\ell}})$ is close to some literal in $\{x_{\tau(\ell)}, \overline{x_{\tau(\ell)}}\}$, with respect to the uniform distribution.*

This is done using the procedure **UniformJunta** in Lemma 4.

If f is k-junta then, by Fact 1 and 2, w.h.p., it passes this test (does not output reject). This is Lemma 7. If the algorithm does not pass this test, it rejects. If f is not k-junta and it passes this test, then the statement in Fact 4 is true. This is proved in Lemma 11.

Consider now steps 18-28. First, let us consider a function f that is ϵ-far from every k-junta with respect to $\mathcal{D}$. Let $J = \{\tau(\ell) \mid \ell \in I\}$ where $\tau(\ell)$ is as defined in Fact 4. Since by Fact 3, w.h.p., $f(x_X \circ 0_{\overline{X}})$ is $\epsilon/2$-far from every k-junta with respect to $\mathcal{D}$ and $|J| = |I| \leq k$, by Lemma 5, w.h.p.,

$$\mathbf{Pr}_{y\in U, x\in\mathcal{D}}[f(x_X \circ 0_{\overline{X}}) \neq f(x_J \circ y_{X\setminus J} \circ 0_{\overline{X}})] \geq \epsilon/2.$$

So we need to test whether $f(x_X \circ 0_{\overline{X}})$ is $\epsilon/2$-far from $f(x_J \circ y_{X\setminus J} \circ 0_{\overline{X}})$ (those are equal in the case when f is k-Junta). This is the last test we would like to do but the problem is that we do not know J, so we cannot use this test as is. So we change it, as is done in [35], to an equivalent test as follows

$$\mathbf{Pr}_{z\in U, x\in\mathcal{D}}[f(x_X \circ 0_{\overline{X}}) \neq f((x_X + z_X) \circ 0_{\overline{X}}) \mid z_J = 0_J] \geq \epsilon/2.$$

To be able to draw uniformly random z_X with $z_J = 0_J$, we use Fact 4, that is, the fact that each $f(x_{X_\ell} \circ v^{(\ell)}_{\overline{X_\ell}})$ is close to one of the literals in $\{x_{\tau(\ell)}, \overline{x_{\tau(\ell)}}\}$. For every $\ell \in I$, the algorithm draws uniformly random $w := z_{X_\ell}$ and then using the fact that $f(x_{X_\ell} \circ v^{(\ell)}_{\overline{X_\ell}})$ is close to one of the literals in $\{x_{\tau(\ell)}, \overline{x_{\tau(\ell)}}\}$ where $\tau(\ell) \in X_\ell$ the algorithm tests in which set $Y_{\ell,0} := \{j \in X_\ell |\ w_j = 0\}$ or $Y_{\ell,1} := \{j \in X_\ell |\ w_j = 1\}$ the index $\tau(\ell)$ falls. If $\tau(\ell) \in Y_{\ell,0}$ then the entry $\tau(\ell)$ in z_{X_ℓ} is zero and if $\tau(\ell) \in Y_{\ell,1}$ then the entry $\tau(\ell)$ in z_{X_ℓ} is one. In the latter case, the algorithm replaces z_{X_ℓ} with $\overline{z_{X_\ell}}$ (negation of each entry in z_{X_ℓ}) which is also uniformly random. This gives a random uniform z_{X_ℓ} with $z_{\tau(\ell)} = 0$. We do that for every $\ell \in I$ and get a random uniform z with $z_J = 0$. This is proved in Lemma 12. Then the algorithm rejects if $f(x_X \circ 0_{\overline{X}}) \neq f((x_X + z_X) \circ 0_{\overline{X}})$. If $f(x_X \circ 0_{\overline{X}})$ is $\epsilon/2$-far from every

k-junta then, by Lemma 5, $f(x_X \circ 0_{\overline{X}})$ is $\epsilon/2$-far from $f(x_J \circ y_{X\setminus J} \circ 0_{\overline{X}})$, and the algorithm, with one test, rejects with probability at least $\epsilon/2$. Therefore, by repeating this test $O(1/\epsilon)$ times the algorithm rejects w.h.p. This is proved in Lemma 13.

Now we consider f that is k-junta. Obviously, if f is k-junta then $f(x_X \circ 0_{\overline{X}}) = f((x_X + z_X) \circ 0_{\overline{X}})$ when $z_J = 0$ and the algorithm accepts. This is because $x(J)$ are the relevant variables in $f(x_X \circ 0_{\overline{X}})$. This is proved in Lemma 8.

3.2 The algorithm for k-Junta

In this subsection, we show that if the target function f is k-junta then the algorithm accepts with probability at least $2/3$.

We first prove

▶ **Lemma 6.** *Consider steps 1-2 in the algorithm. If f is a k-junta then, with probability at least $2/3$, for each $i \in [r]$, the set $x(X_i) = \{x_j | j \in X_i\}$ contains at most one relevant variable of f.*

Proof. Let x_{i_1} and x_{i_2} be two relevant variables in f. The probability that x_{i_1} and x_{i_2} are in the same set is equal to $1/r$. By the union bound, it follows that the probability that some relevant variables x_{i_1} and x_{i_2} in f are in the same set is at most $\binom{k}{2}/r \leq 1/3$. ◀

We now show that w.h.p. the algorithm reaches the final test in the algorithm

▶ **Lemma 7.** *If f is k-junta and each $x(X_i)$ contains at most one relevant variable of f then*
1. *Each $x(X_i)$, $i \in I$, contains exactly one relevant variable.*
2. *The algorithm reaches step 18*

Proof. By Lemma 3 and steps 7-9, for $\ell \in I$, $f(v^{(\ell)}) \neq f(0_{X_\ell} \circ v^{(\ell)}_{\overline{X_\ell}})$ and therefore $x(X_\ell)$ contains exactly one relevant variable. Thus, for every $\ell \in I$, $f(x_{X_\ell} \circ v^{(\ell)}_{\overline{X_\ell}})$ is a literal.

If the algorithm does not reach step 18, then it either halts in step 10, 15 or 17. If it halts in step 10 then $|I| > k$ and therefore, by Lemma 2, f contains more than k relevant variables and then it is not k-Junta. If it halts in step 15 then, by Lemma 4, for some X_ℓ, $\ell \in I$, $f(x_{X_\ell} \circ v^{(\ell)}_{\overline{X_\ell}})$ is not 1-Junta (literal or constant function) and therefore X_ℓ contains at least two relevant variables. If it halts in step 17, then $f(b_{X_\ell} \circ v^{(\ell)}_{\overline{X_\ell}}) = f(\overline{b_{X_\ell}} \circ v^{(\ell)}_{\overline{X_\ell}})$ and then $f(x_{X_\ell} \circ v^{(\ell)}_{\overline{X_\ell}})$ is not a literal. In all cases we get a contradiction. ◀

We now give two Lemmas that show that, with probability at least $2/3$, the algorithm accepts k-junta.

▶ **Lemma 8.** *If f is k-Junta and each $x(X_i)$ contains at most one relevant variable of f then the algorithm outputs "accept".*

Proof. By Lemma 7, the algorithm reaches step 18. We now show that it reaches step 29. Now we need to show that the algorithm does not halt in step 25 or 28.

Since $Y_{\ell,0}, Y_{\ell,1}$ is a partition of X_ℓ, $\ell \in I$ and X_ℓ contains exactly one relevant variable in $x(X_\ell)$ of f, this variable is either in $x(Y_{\ell,0})$ or in $x(Y_{\ell,1})$ but not in both. Suppose w.l.o.g. it is in $x(Y_{\ell,0})$ and not in $x(Y_{\ell,1})$. Then $f(x_{Y_{\ell,0}} \circ b_{Y_{\ell,1}} \circ v^{(\ell)}_{\overline{X_\ell}})$ is a literal and $f(x_{Y_{\ell,1}} \circ b_{Y_{\ell,0}} \circ v^{(\ell)}_{\overline{X_\ell}})$ is a constant function. This implies that for any b, $f(b_{Y_{\ell,0}} \circ b_{Y_{\ell,1}} \circ v^{(\ell)}_{\overline{X_\ell}}) \neq f(\overline{b_{Y_{\ell,0}}} \circ b_{Y_{\ell,1}} \circ v^{(\ell)}_{\overline{X_\ell}})$ and $f(b_{Y_{\ell,1}} \circ b_{Y_{\ell,0}} \circ v^{(\ell)}_{\overline{X_\ell}}) = f(\overline{b_{Y_{\ell,1}}} \circ b_{Y_{\ell,0}} \circ v^{(\ell)}_{\overline{X_\ell}})$. Therefore, $G_{\ell,0} = h$ and $G_{\ell,1} = 0$. Thus the algorithm does not halt in step 25.

Algorithm Simple$\mathcal{D}k$−Junta$(f, \mathcal{D}, \epsilon)$
Input: Oracle that accesses a Boolean function f and
oracle that draws a random $x \in \{0,1\}^n$ according to the distribution $\mathcal{D}$.
Output: Either "accept" or "reject"

Partition $[n]$ into r sets
1. Set $r = 2k^2$.
2. Choose uniformly at random a partition $X_1, X_2, \ldots, X_r$ of $[n]$

Find a close function and relevant sets
3. Set $X = \emptyset$; $I = \emptyset$; $t(X) = 0$
4. Repeat $M = 2k\ln(15k)/\epsilon$ times
5. Choose $u \in \mathcal{D}$.
6. $t(X) \leftarrow t(X) + 1$
7. If $f(u_X \circ 0_{\overline{X}}) \neq f(u)$ then
8. Binary search: find a new relevant set X_ℓ; $X \leftarrow X \cup X_\ell$; $I \leftarrow I \cup \{\ell\}$
9. and a string $v^{(\ell)} \in \{0,1\}^n$ such that $f(v^{(\ell)}) \neq f(0_{X_\ell} \circ v^{(\ell)}_{\overline{X_\ell}})$.
10. If $|I| > k$ then output "reject" and halt.
11. $t(X) = 0$.
12. If $t(X) = 2\ln(15k)/\epsilon$ then Goto 13.

Tests if each relevant set is close to a literal
13. For every $\ell \in I$ do
14. If **UniformJunta**$(f(x_{X_\ell} \circ v^{(\ell)}_{\overline{X_\ell}}), 1, 1/30, 1/15)$="reject"
15. then output "reject" and halt
16. Choose $b \in U$
17. If $f(b_{X_\ell} \circ v^{(\ell)}_{\overline{X_\ell}}) = f(\overline{b_{X_\ell}} \circ v^{(\ell)}_{\overline{X_\ell}})$ then output "reject" and halt

The final test of Lemma 5
18. Repeat $M' = (2\ln 15)/\epsilon$ times
19. Choose $w \in U$; $z = 0_{\overline{X}}$
20. For every $\ell \in I$ do
21. Set $Y_{\ell,\xi} = \{j \in X_\ell | w_j = \xi\}$ for $\xi \in \{0,1\}$.
22. Set $G_{\ell,0} = G_{\ell,1} = 0$;
23. Repeat $h = \ln(15M'k)/\ln(4/3)$ times
24. Choose $b \in U$;
If $f(b_{Y_{\ell,0}} \circ b_{Y_{\ell,1}} \circ v^{(\ell)}_{\overline{X_\ell}}) \neq f(\overline{b_{Y_{\ell,0}}} \circ b_{Y_{\ell,1}} \circ v^{(\ell)}_{\overline{X_\ell}})$ then $G_{\ell,0} \leftarrow G_{\ell,0} + 1$
If $f(b_{Y_{\ell,1}} \circ b_{Y_{\ell,0}} \circ v^{(\ell)}_{\overline{X_\ell}}) \neq f(\overline{b_{Y_{\ell,1}}} \circ b_{Y_{\ell,0}} \circ v^{(\ell)}_{\overline{X_\ell}})$ then $G_{\ell,1} \leftarrow G_{\ell,1} + 1$
25. If $(\{G_{\ell,0}, G_{\ell,1}\} \neq \{0, h\})$ then output "reject" and halt
26. If $G_{\ell,0} = h$ then $z \leftarrow z \circ w_{X_\ell}$ else $z \leftarrow z \circ \overline{w_{X_\ell}}$
27. Choose $u \in \mathcal{D}$
28. If $f(u_X \circ 0_{\overline{X}}) \neq f((u_X + z_X) \circ 0_{\overline{X}})$ then output "reject" and halt.
29. Output "accept"

Figure 1 A two-sided distribution-free adaptive algorithm for ϵ-testing k-junta.

Now for every X_ℓ, $\ell \in I$, let $\tau(\ell) \in X_\ell$ be such that $f(x_{X_\ell} \circ v^{(\ell)}_{\overline{X_\ell}}) \in \{x_{\tau(\ell)}, \overline{x_{\tau(\ell)}}\}$. If $\tau(\ell) \in Y_{\ell,0}$ then $G_{\ell,0} = h$ and then by step 26, $z_{\tau(\ell)} = w_{\tau(\ell)} = 0$. If $\tau(\ell) \in Y_{\ell,1}$ then $G_{\ell,1} = h$ and then $z_{\tau(\ell)} = \overline{w_{\tau(\ell)}} = 0$. Therefore for every relevant variable $x_{\tau(\ell)}$ in $\hat{f} = f(x_X \circ 0_{\overline{X}})$ we have $z_{\tau(\ell)} = 0$ which implies that $f(u_X \circ 0_{\overline{X}}) = f((u_X + z_X) \circ 0_{\overline{X}})$ and therefore the algorithm does not halt in step 28. ◀

▶ **Lemma 9.** *If f is k-Junta then the algorithm outputs "accept" with probability at least* $2/3$ *.*

Proof. The result follows from Lemma 6 and Lemma 8. ◀

3.3 The Algorithm for ϵ-Far Functions

In this subsection, we prove that if f is ϵ-far from every k-junta then the algorithm rejects with probability at least $2/3$.

The first lemma shows that, w.h.p., $f(u_X \circ 0_{\overline{X}})$ is $\epsilon/2$-close to f.

▶ **Lemma 10.** *If the algorithm reaches step 13 then $t(X) = 2\ln(15k)/\epsilon$ and $|I| \le k$. If*

$$\boldsymbol{Pr}_{u\in\mathcal{D}}[f(u_X \circ 0_{\overline{X}}) \neq f(u)] \ge \epsilon/2$$

then the algorithm reaches step 13 with probability at most $1/15$*.*

Proof. The algorithm does not reaches step 13 if and only if it halts in step 10 and then $|I| > k$. The size of I is increased by one each time the condition, $f(u_X \circ 0_{\overline{X}}) \neq f(u)$, in step 7, is true. Therefore, if the algorithm reaches step 13 then the condition in step 7 was true at most k times and $|I| \le k$. Then steps 8-11 are executed at most k times. Thus, $t()$ is updated to 0 at most k times. The loop 5-12 is repeated M times and $t()$ is updated to 0 at most k times and therefore there is X for which $t(X) = M/k = 2\ln(15k)/\epsilon$. This implies that when the algorithm reaches step 13, we have $t(X) = 2\ln(15k)/\epsilon$.

The probability that the algorithm reaches step 13 with $\mathbf{Pr}_{u\in\mathcal{D}}[f(u_X \circ 0_{\overline{X}}) \neq f(u)] > \epsilon/2$ is the probability that for one (of the at most k) X', $\mathbf{Pr}_{u\in\mathcal{D}}[f(u_{X'} \circ 0_{\overline{X'}}) \neq f(u)] > \epsilon/2$ and $t(X') = 2\ln(15k)/\epsilon$. By the union bound, this probability is less than

$$k\left(1 - \frac{\epsilon}{2}\right)^{2\ln(15k)/\epsilon} = \frac{1}{15}.$$ ◀

In the following lemma we show that, w.h.p, each $f(x_{X_\ell} \circ v^{(\ell)}_{\overline{X_\ell}})$ is close to a literal.

▶ **Lemma 11.** *Consider steps 13-15. If for some $\ell \in I$, $f(x_{X_\ell} \circ v^{(\ell)}_{\overline{X_\ell}})$ is $(1/30)$-far from every literal with respect to the uniform distribution then, with probability at least $1 - (2/15)$, the algorithm rejects.*

Proof. If $f(x_{X_\ell} \circ v^{(\ell)}_{\overline{X_\ell}})$ is $(1/30)$-far from every literal with respect to the uniform distribution then it is either (case 1) $(1/30)$-far from every 1-Junta (literal or constant) or (case 2) $(1/30)$-far from every literal and $(1/30)$-close to 0-Junta. In case 1, by Lemma 4, with probability at least $1 - (1/15)$, **UniformJunta** $(f(x_{X_\ell} \circ v^{(\ell)}_{\overline{X_\ell}}), 1, 1/30, 1/15)$ = "reject" and then the algorithm rejects. In case 2, if $f(x_{X_\ell} \circ v^{(\ell)}_{\overline{X_\ell}})$ is $1/30$-close to some 0-Junta then it is either $(1/30)$-close to 0 or $(1/30)$-close to 1. Suppose it is $(1/30)$-close to 0. Let b be a random uniform string generated in steps 16. Then $\overline{b}$ is random uniform and for $g(x) = f(x_{X_\ell} \circ v^{(\ell)}_{\overline{X_\ell}})$ we have

$$\begin{aligned}\mathbf{Pr}[\text{The algorithm does not reject}] &= \mathbf{Pr}\left[g(b) \neq g(\bar{b})\right] \\ &= \mathbf{Pr}[g(b) = 1 \wedge g(\bar{b}) = 0] + \mathbf{Pr}[g(b) = 0 \wedge g(\bar{b}) = 1] \\ &\leq \mathbf{Pr}[g(b) = 1] + \mathbf{Pr}[g(\bar{b}) = 1] \\ &\leq \frac{1}{15}.\end{aligned}$$

By the union bound the result follows. ◀

In the next lemma we prove that, w.h.p, the string z generated in steps 19-26 satisfies $z_J = 0$ where $x(J)$ are relevant variables of $f(u_X \circ 0_{\overline{X}})$.

▶ **Lemma 12.** *Consider steps 19-26. If for every $\ell \in I$ the function $f(x_{X_\ell} \circ v^{(\ell)}_{\overline{X_\ell}})$ is $(1/30)$-close to a literal in $\{x_{\tau(\ell)}, \bar{x}_{\tau(\ell)}\}$ with respect to the uniform distribution, where $\tau(\ell) \in X_\ell$, and $\{G_{\ell,0}, G_{\ell,1}\} = \{0, h\}$ then, with probability at least $1 - k(3/4)^h$, we have: For every $\ell \in I$, $z_{\tau(\ell)} = 0$.*

Proof. Fix some ℓ. Suppose $f(x_{X_\ell} \circ v^{(\ell)}_{\overline{X_\ell}})$ is $(1/30)$-close to $x_{\tau(\ell)}$ with respect to the uniform distribution. The case when it is $(1/30)$-close to $\overline{x_{\tau(\ell)}}$ is similar. Since $X_\ell = Y_{\ell,0} \cup Y_{\ell,1}$ and $Y_{\ell,0} \cap Y_{\ell,1} = \emptyset$ we have that $\tau(\ell) \in Y_{\ell,0}$ or $\tau(\ell) \in Y_{\ell,1}$, but not both. Suppose $\tau(\ell) \in Y_{\ell,0}$. The case where $\tau(\ell) \in Y_{\ell,1}$ is similar. Define the random variable $Z(x_{X_\ell}) = 1$ if $f(x_{X_\ell} \circ v^{(\ell)}_{\overline{X_\ell}}) \neq x_{\tau(\ell)}$ and $Z(x_{X_\ell}) = 0$ otherwise. Then

$$\mathbf{E}_{x_{X_\ell} \in U}[Z(x_{X_\ell})] \leq \frac{1}{30}.$$

Therefore

$$\mathbf{E}_{x_{Y_{\ell,1}} \in U} \mathbf{E}_{x_{Y_{\ell,0}} \in U}[Z(x_{Y_{\ell,0}} \circ x_{Y_{\ell,1}})] \leq \frac{1}{30}$$

and by Markov's bound

$$\mathbf{Pr}_{x_{Y_{\ell,1}} \in U}\left[\mathbf{E}_{x_{Y_{\ell,0}} \in U}[Z(x_{Y_{\ell,0}} \circ x_{Y_{\ell,1}})] \geq \frac{2}{15}\right] \leq \frac{1}{4}.$$

That is, for a random uniform string $b \in \{0,1\}^n$, with probability at least $3/4$, $f(x_{Y_{\ell,0}} \circ b_{Y_{\ell,1}} \circ v^{(\ell)}_{\overline{X_\ell}})$ is $(2/15)$-close to $x_{\tau(\ell)}$ with respect to the uniform distribution. Now, given that $f(x_{Y_{\ell,0}} \circ b_{Y_{\ell,1}} \circ v^{(\ell)}_{\overline{X_\ell}})$ is $(2/15)$-close to $x_{\tau(\ell)}$ with respect to the uniform distribution the probability that $G_{\ell,0} = 0$ is the probability that $f(b_{Y_{\ell,0}} \circ b_{Y_{\ell,1}} \circ v^{(\ell)}_{\overline{X_\ell}}) = f(\overline{b_{Y_{\ell,0}}} \circ b_{Y_{\ell,1}} \circ v^{(\ell)}_{\overline{X_\ell}})$ for h random uniform strings $b \in \{0,1\}^n$. Let $b^{(1)}, \ldots, b^{(h)}$ be h random uniform strings in $\{0,1\}^n$, $V(b)$ be the event $f(b_{Y_{\ell,0}} \circ b_{Y_{\ell,1}} \circ v^{(\ell)}_{\overline{X_\ell}}) = f(\overline{b_{Y_{\ell,0}}} \circ b_{Y_{\ell,1}} \circ v^{(\ell)}_{\overline{X_\ell}})$ and A the event that $f(x_{Y_{\ell,0}} \circ b_{Y_{\ell,1}} \circ v^{(\ell)}_{\overline{X_\ell}})$ is $(2/15)$-close to $x_{\tau(\ell)}$ with respect to the uniform distribution. Let $g(x_{Y_{\ell,0}}) = f(x_{Y_{\ell,0}} \circ b_{Y_{\ell,1}} \circ v^{(\ell)}_{\overline{X_\ell}})$. Then

$$\begin{aligned}\mathbf{Pr}[V(b)|A] &= \mathbf{Pr}[g(b_{Y_{\ell,0}}) = g(\overline{b_{Y_{\ell,0}}})|A] \\ &= \mathbf{Pr}[(g(b_{Y_{\ell,0}}) = b_{\tau(\ell)} \wedge g(\overline{b_{Y_{\ell,0}}}) = b_{\tau(\ell)}) \vee \\ &\qquad (g(b_{Y_{\ell,0}}) = \overline{b_{\tau(\ell)}} \wedge g(\overline{b_{Y_{\ell,0}}}) = \overline{b_{\tau(\ell)}})|A] \\ &\leq \mathbf{Pr}[g(\overline{b_{Y_{\ell,0}}}) \neq \overline{b_{\tau(\ell)}} \vee g(b_{Y_{\ell,0}}) \neq b_{\tau(\ell)})|A] \\ &\leq \mathbf{Pr}[g(\overline{b_{Y_{\ell,0}}}) \neq \overline{b_{\tau(\ell)}}|A] + \mathbf{Pr}[g(b_{Y_{\ell,0}}) \neq b_{\tau(\ell)})|A] \leq \frac{4}{15}.\end{aligned}$$

Since $\tau(\ell) \in Y_{\ell,0}$, we have $w_{\tau(\ell)} = 0$. Therefore, by step 26 and since $\tau(\ell) \in X_\ell$,

$$\begin{aligned}\mathbf{Pr}[z_{\tau(\ell)} = 1] &= \mathbf{Pr}[G_{\ell,0} = 0 \wedge G_{\ell,1} = h] \\ &\le \mathbf{Pr}[G_{\ell,0} = 0] = \mathbf{Pr}[(\forall j \in [h])V(b^{(j)})] \\ &= (\mathbf{Pr}[V(b)])^h \le \left(\mathbf{Pr}[V(b)|A] + \mathbf{Pr}[\overline{A}]\right)^h \le (4/15 + 1/4)^h \le (3/4)^h\end{aligned}$$

Therefore, the probability that $z_{\tau(\ell)} = 1$ for some $\ell \in I$ is at most $k(3/4)^h$. ◀

We now show that w.h.p the algorithm reject if f is ϵ-far from every k-junta

▶ **Lemma 13.** *If f is ϵ-far from every k-junta with respect to $\mathcal{D}$ then, with probability at least $2/3$, the algorithm outputs "reject".*

Proof. If the algorithm stops in step 10 then we are done. Therefore we may assume that

$$|I| \le k. \tag{1}$$

By Lemma 10, if $\mathbf{Pr}_{u\in\mathcal{D}}[f(u_X \circ 0_{\overline{X}}) \ne f(u)] \ge \epsilon/2$ then, with probability at most $1/15$, the algorithm reaches step 13. So we may assume that (failure probability $1/15$)

$$\mathbf{Pr}_{u\in\mathcal{D}}[f(u_X \circ 0_{\overline{X}}) \ne f(u)] \le \epsilon/2. \tag{2}$$

Since f is ϵ-far from every k-junta with respect to $\mathcal{D}$ and $f(x_X \circ 0_{\overline{X}})$ is $\epsilon/2$-close to f with respect to $\mathcal{D}$ we have $f(x_X \circ 0_{\overline{X}})$ is $(\epsilon/2)$-far from every k-junta with respect to $\mathcal{D}$. Therefore, by Lemma 5,

$$\mathop{\mathbf{Pr}}_{u\in\mathcal{D},y\in U}[f(u_X \circ 0_{\overline{X}}) = f(u_I \circ y_{X\setminus I} \circ 0_{\overline{X}})] \ge 1 - \frac{\epsilon}{2}. \tag{3}$$

By Lemma 11, if some $f(x_{X_\ell} \circ v^{(\ell)}_{\overline{X_\ell}})$ is $(1/30)$-far from any literal with respect to the uniform distribution then, with probability at least $1 - (2/15)$, the algorithm rejects. So we may assume (failure probability $2/15$) that every $f(x_{X_\ell} \circ v^{(\ell)}_{\overline{X_\ell}})$ is $(1/30)$-close to some $x_{\tau(\ell)}$ or $\overline{x_{\tau(\ell)}}$ with respect to the uniform distribution, where $\tau(\ell) \in X_\ell$.

Let $z^{(1)}, \ldots, z^{(M')}$ be the strings generated in step 26. By Lemma 12, with probability at least $1 - M'k(3/4)^h \ge 1 - (1/15)$, every $z^{(i)}$ generated in step 26 satisfies $z^{(i)}_{\tau(\ell)} = 0$ for all $\ell \in I$. Also, since the distribution of w_{X_ℓ} and $\overline{w_{X_\ell}}$ is uniform, the distribution of $z^{(i)}_{X\setminus I}$ and $u_{X\setminus I} + z^{(i)}_{X\setminus I}$ is uniform. We now assume (failure probability $1/15$) that $z^{(i)}_I = 0$ for all i. Therefore, by (3),

$$\begin{aligned}\mathop{\mathbf{Pr}}_{u\in\mathcal{D},z^{(i)}_{X\setminus I}\in U}[(\forall i) f(u_X \circ 0_{\overline{X}}) &= f((u_X + z^{(i)}_X) \circ 0_{\overline{X}})] \\ &= \left(\mathop{\mathbf{Pr}}_{u\in\mathcal{D},z^{(1)}_{X\setminus I}\in U}[f(u_X \circ 0_{\overline{X}}) = f((u_X + z^{(1)}_X) \circ 0_{\overline{X}})]\right)^{M'} \\ &= \left(\mathop{\mathbf{Pr}}_{u\in\mathcal{D},y\in U}[f(u_X \circ 0_{\overline{X}}) = f(u_I \circ y_{X\setminus I} \circ 0_{\overline{X}})]\right)^{M'} \\ &\le (1-\epsilon/2)^{M'} \le \frac{1}{15}.\end{aligned}$$

Therefore, the failure probability of an output "reject" is at most $1/15 + 2/15 + 1/15 + 1/15 = 1/3$. ◀

3.4 The Query Complexity of the Algorithm

In this section we show that

▶ **Lemma 14.** *The query complexity of the algorithm is*

$$\tilde{O}\left(\frac{k}{\epsilon}\right).$$

Proof. The condition in step 7 requires two queries and is executed at most $M = 2k\ln(15k)/\epsilon$ times. This is $2M = O((k\log k)/\epsilon)$ queries. Steps 8 is executed at most $k+1$ times. This is because each time it is executed, the value of $|I|$ is increased by one, and when $|I| = k+1$ the algorithm rejects. By Lemma 3, to find a new relevant set the algorithm makes $O(\log r) = O(\log k)$ queries. This is $O(k\log k)$ queries. Steps 14 and 17 are executed $|I| \leq k$ times, and by Lemma 4, the total number of queries made is $O(1/(1/30)\log(15))k+2k = O(k)$.

The final test in the algorithm is repeated $M' = (2\ln 15)/\epsilon$ times (step 18) and each time, and for each $\ell \in I$, (step 20) it repeats h times (step 23) two conditions that takes 2 queries each (step 24). This takes $4M'kh = O((k/\epsilon)\ln(k/\epsilon))$ queries. The number of queries in step 28 is $2M' = O(1/\epsilon)$. Therefore the total number of queries is

$$O\left(\frac{k}{\epsilon}\ln\frac{k}{\epsilon}\right).$$ ◀

4 Open Problems

In this paper we proved that for any $\epsilon > 0$, there is a two-sided distribution-free adaptive algorithm for ϵ-testing k-junta that makes $\tilde{O}(k/\epsilon)$ queries. It is also interesting to find a one-sided distribution-free adaptive algorithm with such query complexity.

Chen et al. [35] proved the lower bound $\Omega(2^{k/3})$ for any non-adaptive (one round) algorithm. What is the minimal number rounds one needs to get $poly(k/\epsilon)$ query complexity? Can $O(1)$-round algorithms solve the problem with $poly(k/\epsilon)$ queries?

In the uniform distribution framework, where the distance between two functions is measured with respect to the uniform distribution Blais in [5] gave a non-adaptive algorithm that makes $\tilde{O}(k^{3/2})/\epsilon$ queries and in [6] an adaptive algorithm that makes $O(k\log k + k/\epsilon)$ queries. On the lower bounds side, Sağlam in [43] gave an $\Omega(k\log k)$ lower bound for adaptive testing and Chen et al. [17] gave an $\tilde{\Omega}(k^{3/2})/\epsilon$ lower bound for the non-adaptive testing. Thus in both the adaptive and non-adaptive uniform distribution settings, the query complexity of k-junta testing has now been pinned down to within logarithmic factors. It is interesting to study $O(1)$-round algorithms. For example, what is the query complexity for 2-round algorithm.

References

1 Noga Alon, Tali Kaufman, Michael Krivelevich, Simon Litsyn, and Dana Ron. Testing Reed-Muller codes. *IEEE Trans. Information Theory*, 51(11):4032–4039, 2005. doi:10.1109/TIT.2005.856958.

2 Roksana Baleshzar, Meiram Murzabulatov, Ramesh Krishnan S. Pallavoor, and Sofya Raskhodnikova. Testing Unateness of Real-Valued Functions. *CoRR*, abs/1608.07652, 2016. arXiv:1608.07652.

3 Aleksandrs Belovs and Eric Blais. A polynomial lower bound for testing monotonicity. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 1021–1032, 2016. doi:10.1145/2897518.2897567.

4 Arnab Bhattacharyya, Swastik Kopparty, Grant Schoenebeck, Madhu Sudan, and David Zuckerman. Optimal Testing of Reed-Muller Codes. In *Property Testing - Current Research and Surveys*, pages 269–275. Springer, 2010. doi:10.1007/978-3-642-16367-8_19.

5 Eric Blais. Improved Bounds for Testing Juntas. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques, 11th International Workshop, APPROX 2008, and 12th International Workshop, RANDOM 2008, Boston, MA, USA, August 25-27, 2008. Proceedings*, pages 317–330, 2008. doi:10.1007/978-3-540-85363-3_26.

6 Eric Blais. Testing juntas nearly optimally. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pages 151–158, 2009. doi:10.1145/1536414.1536437.

7 Eric Blais, Joshua Brody, and Kevin Matulef. Property Testing Lower Bounds via Communication Complexity. In *Proceedings of the 26th Annual IEEE Conference on Computational Complexity, CCC 2011, San Jose, California, USA, June 8-10, 2011*, pages 210–220, 2011. doi:10.1109/CCC.2011.31.

8 Eric Blais and Daniel M. Kane. Tight Bounds for Testing k-Linearity. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques - 15th International Workshop, APPROX 2012, and 16th International Workshop, RANDOM 2012, Cambridge, MA, USA, August 15-17, 2012. Proceedings*, pages 435–446, 2012. doi:10.1007/978-3-642-32512-0_37.

9 Avrim Blum. Learning a Function of r Relevant Variables. In *Computational Learning Theory and Kernel Machines, 16th Annual Conference on Computational Learning Theory and 7th Kernel Workshop, COLT/Kernel 2003, Washington, DC, USA, August 24-27, 2003, Proceedings*, pages 731–733, 2003. doi:10.1007/978-3-540-45167-9_54.

10 Avrim Blum and Pat Langley. Selection of Relevant Features and Examples in Machine Learning. *Artif. Intell.*, 97(1-2):245–271, 1997. doi:10.1016/S0004-3702(97)00063-5.

11 Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-Testing/Correcting with Applications to Numerical Problems. *J. Comput. Syst. Sci.*, 47(3):549–595, 1993. doi:10.1016/0022-0000(93)90044-W.

12 Nader H. Bshouty and Areej Costa. Exact learning of juntas from membership queries. *Theor. Comput. Sci.*, 742:82–97, 2018. doi:10.1016/j.tcs.2017.12.032.

13 Deeparnab Chakrabarty and C. Seshadhri. A $o(n)$ monotonicity tester for boolean functions over the hypercube. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 411–418, 2013. doi:10.1145/2488608.2488660.

14 Deeparnab Chakrabarty and C. Seshadhri. A $\tilde{O}(n)$ Non-Adaptive Tester for Unateness. *CoRR*, abs/1608.06980, 2016. arXiv:1608.06980.

15 Xi Chen, Anindya De, Rocco A. Servedio, and Li-Yang Tan. Boolean Function Monotonicity Testing Requires (Almost) $n^{1/2}$ Non-adaptive Queries. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 519–528, 2015. doi:10.1145/2746539.2746570.

16 Xi Chen, Rocco A. Servedio, and Li-Yang Tan. New algorithms and lower bounds for monotonicity testing. *CoRR*, abs/1412.5655, 2014. arXiv:1412.5655.

17 Xi Chen, Rocco A. Servedio, Li-Yang Tan, Erik Waingarten, and Jinyu Xie. Settling the Query Complexity of Non-Adaptive Junta Testing. In *32nd Computational Complexity Conference, CCC 2017, July 6-9, 2017, Riga, Latvia*, pages 26:1–26:19, 2017. doi:10.4230/LIPIcs.CCC.2017.26.

18 Xi Chen, Erik Waingarten, and Jinyu Xie. Beyond Talagrand functions: new lower bounds for testing monotonicity and unateness. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 523–536, 2017. doi:10.1145/3055399.3055461.

19 Xi Chen, Erik Waingarten, and Jinyu Xie. Boolean Unateness Testing with $\widetilde{O}(n^{3/4})$ Adaptive Queries. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 868–879, 2017. doi:10.1109/FOCS.2017.85.

20 Xi Chen and Jinyu Xie. Tight Bounds for the Distribution-Free Testing of Monotone Conjunctions. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 54–71, 2016. doi:10.1137/1.9781611974331.ch5.

21 Hana Chockler and Dan Gutfreund. A lower bound for testing juntas. *Inf. Process. Lett.*, 90(6):301–305, 2004. doi:10.1016/j.ipl.2004.01.023.

22 Ilias Diakonikolas, Homin K. Lee, Kevin Matulef, Krzysztof Onak, Ronitt Rubinfeld, Rocco A. Servedio, and Andrew Wan. Testing for Concise Representations. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2007), October 20-23, 2007, Providence, RI, USA, Proceedings*, pages 549–558, 2007. doi:10.1109/FOCS.2007.32.

23 Elya Dolev and Dana Ron. Distribution-Free Testing for Monomials with a Sublinear Number of Queries. *Theory of Computing*, 7(1):155–176, 2011. doi:10.4086/toc.2011.v007a011.

24 Eldar Fischer, Eric Lehman, Ilan Newman, Sofya Raskhodnikova, Ronitt Rubinfeld, and Alex Samorodnitsky. Monotonicity testing over general poset domains. In *Proceedings on 34th Annual ACM Symposium on Theory of Computing, May 19-21, 2002, Montréal, Québec, Canada*, pages 474–483, 2002. doi:10.1145/509907.509977.

25 Dana Glasner and Rocco A. Servedio. Distribution-Free Testing Lower Bound for Basic Boolean Functions. *Theory of Computing*, 5(1):191–216, 2009. doi:10.4086/toc.2009.v005a010.

26 Oded Goldreich, editor. *Property Testing - Current Research and Surveys*, volume 6390 of *Lecture Notes in Computer Science*. Springer, 2010. doi:10.1007/978-3-642-16367-8.

27 Oded Goldreich, Shafi Goldwasser, Eric Lehman, Dana Ron, and Alex Samorodnitsky. Testing Monotonicity. *Combinatorica*, 20(3):301–337, 2000. doi:10.1007/s004930070011.

28 Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property Testing and its Connection to Learning and Approximation. *J. ACM*, 45(4):653–750, 1998. doi:10.1145/285055.285060.

29 Parikshit Gopalan, Ryan O'Donnell, Rocco A. Servedio, Amir Shpilka, and Karl Wimmer. Testing Fourier Dimensionality and Sparsity. *SIAM J. Comput.*, 40(4):1075–1100, 2011. doi:10.1137/100785429.

30 David Guijarro, Jun Tarui, and Tatsuie Tsukiji. Finding Relevant Variables in PAC Model with Membership Queries. In *Algorithmic Learning Theory, 10th International Conference, ALT '99, Tokyo, Japan, December 6-8, 1999, Proceedings*, page 313, 1999. doi:10.1007/3-540-46769-6_26.

31 Shirley Halevy and Eyal Kushilevitz. Distribution-Free Property-Testing. *SIAM J. Comput.*, 37(4):1107–1138, 2007. doi:10.1137/050645804.

32 Subhash Khot, Dor Minzer, and Muli Safra. On Monotonicity Testing and Boolean Isoperimetric Type Theorems. In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 52–58, 2015. doi:10.1109/FOCS.2015.13.

33 Subhash Khot and Igor Shinkar. An $\widetilde{O}(n)$ Queries Adaptive Tester for Unateness. *CoRR*, abs/1608.02451, 2016. arXiv:1608.02451.

34 Richard J. Lipton, Evangelos Markakis, Aranyak Mehta, and Nisheeth K. Vishnoi. On the Fourier Spectrum of Symmetric Boolean Functions with Applications to Learning Symmetric Juntas. In *20th Annual IEEE Conference on Computational Complexity (CCC 2005), 11-15 June 2005, San Jose, CA, USA*, pages 112–119, 2005. doi:10.1109/CCC.2005.19.

35 Zhengyang Liu, Xi Chen, Rocco A. Servedio, Ying Sheng, and Jinyu Xie. Distribution-free junta testing. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 749–759, 2018. doi:10.1145/3188745.3188842.

36 Kevin Matulef, Ryan O'Donnell, Ronitt Rubinfeld, and Rocco A. Servedio. Testing ± 1-weight halfspace. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, 12th International Workshop, APPROX 2009, and 13th International Workshop, RANDOM 2009, Berkeley, CA, USA, August 21-23, 2009. Proceedings*, pages 646–657, 2009. doi:10.1007/978-3-642-03685-9_48.

37 Kevin Matulef, Ryan O'Donnell, Ronitt Rubinfeld, and Rocco A. Servedio. Testing Halfspaces. *SIAM J. Comput.*, 39(5):2004–2047, 2010. doi:10.1137/070707890.

38 Elchanan Mossel, Ryan O'Donnell, and Rocco A. Servedio. Learning functions of k relevant variables. *J. Comput. Syst. Sci.*, 69(3):421–434, 2004. doi:10.1016/j.jcss.2004.04.002.

39 Michal Parnas, Dana Ron, and Alex Samorodnitsky. Testing Basic Boolean Formulae. *SIAM J. Discrete Math.*, 16(1):20–46, 2002. URL: http://epubs.siam.org/sam-bin/dbq/article/40744.

40 Dana Ron. Property Testing: A Learning Theory Perspective. *Foundations and Trends in Machine Learning*, 1(3):307–402, 2008. doi:10.1561/2200000004.

41 Dana Ron. Algorithmic and Analysis Techniques in Property Testing. *Foundations and Trends in Theoretical Computer Science*, 5(2):73–205, 2009. doi:10.1561/0400000029.

42 Ronitt Rubinfeld and Madhu Sudan. Robust Characterizations of Polynomials with Applications to Program Testing. *SIAM J. Comput.*, 25(2):252–271, 1996. doi:10.1137/S0097539793255151.

43 Mert Saglam. Near Log-Convexity of Measured Heat in (Discrete) Time and Consequences. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 967–978, 2018. doi:10.1109/FOCS.2018.00095.

UG-Hardness to NP-Hardness by Losing Half

Amey Bhangale
Weizmann Institute of Science, Rehovot, Israel
amey.bhangale@weizmann.ac.il

Subhash Khot
Courant Institute of Mathematical Sciences, New York University, USA
khot@cs.nyu.edu

Abstract

The 2-to-2 Games Theorem of [16, 10, 11, 17] implies that it is NP-hard to distinguish between Unique Games instances with assignment satisfying at least $(\frac{1}{2} - \varepsilon)$ fraction of the constraints *vs.* no assignment satisfying more than ε fraction of the constraints, for every constant $\varepsilon > 0$. We show that the reduction can be transformed in a non-trivial way to give a stronger guarantee in the completeness case: For at least $(\frac{1}{2} - \varepsilon)$ fraction of the vertices on one side, all the constraints associated with them in the Unique Games instance can be satisfied.

We use this guarantee to convert the known UG-hardness results to NP-hardness. We show:

1. Tight inapproximability of approximating independent sets in degree d graphs within a factor of $\Omega\left(\frac{d}{\log^2 d}\right)$, where d is a constant.
2. NP-hardness of approximate the Maximum Acyclic Subgraph problem within a factor of $\frac{2}{3} + \varepsilon$, improving the previous ratio of $\frac{14}{15} + \varepsilon$ by Austrin *et al.* [4].
3. For any predicate $P^{-1}(1) \subseteq [q]^k$ supporting a balanced pairwise independent distribution, given a P-CSP instance with value at least $\frac{1}{2} - \varepsilon$, it is NP-hard to satisfy more than $\frac{|P^{-1}(1)|}{q^k} + \varepsilon$ fraction of constraints.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis

Keywords and phrases NP-hardness, Inapproximability, Unique Games Conjecture

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.3

Funding *Amey Bhangale*: Research supported by Irit Dinur's ERC-CoG grant 772839.
Subhash Khot: Research supported by NSF CCF-1813438, Simons Collaboration on Algorithms and Geometry, and Simons Investigator Award.

Acknowledgements We would like to thank anonymous referees for helpful comments.

1 Introduction

Unique Games Conjecture (UGC) is a central open problem in computer science. It states that for a certain constraint satisfaction problem over a large alphabet, called *Unique Games* (UG), it is NP-hard to decide whether a given instance has an assignment that satisfies almost all the constraints or there is no assignment which satisfies even an ε fraction of the constraints for a very small constant $\varepsilon > 0$.

Since the formulation of the conjecture, it has found interesting connections to tight hardness of approximation result for many optimization problems [14, 15, 19, 23, 13, 18, 20, 21]. One of the most notable implications is the result of Raghavendra [23] which informally can be stated as follows: Assuming the NP-hardness of approximating this single CSP (Unique Games) implies tight hardness for approximating every other constraint satisfaction problem, stated in terms of integrality gap of certain canonical SDP.

Unique Games Conjecture is inspired by the NP-hardness of approximating a problem called Label Cover. A Label Cover instance consists of two sets of variables A and B and a bipartite graph G between them. The variables from A take values from some alphabet

Σ_A and variables from B take values from Σ_B. Every edge e in G has a d-to-1 projection constraint $\pi : \Sigma_A \to \Sigma_B$.[1] For an edge (a, b), a label α to a and a label β to b satisfies the edge iff $\pi(\alpha) = \beta$ where π is a constraint on the edge (a, b). In this language, Unique Games is a Label Cover instance where all the constraints are 1-to-1. An instance is called ε-satisfiable if there exists an assignment $\sigma : A \cup B \to \Sigma_A \cup \Sigma_B$, that satisfies at least ε fraction of the edges in the graph.

A recent series of works [16, 10, 11, 17] implies that for a given Label Cover instance with 2-to-1 projection constraints, it is NP-hard to find an ε-satisfiable assignment even if the instance is $(1-\varepsilon)$-satisfiable for all $\varepsilon > 0$. This directly implies the following inapproximability for Unique Games.

▶ **Theorem 1.** *For every $\varepsilon > 0$, there exists Σ such that for Unique Games instance over Σ, it is NP-hard to distinguish between the following two cases*

- *Yes Case: The instance is $(\frac{1}{2} - \varepsilon)$-satisfiable.*
- *No Case: No assignment satisfies ε fraction of the constraints.*

Although we do not improve upon this theorem in terms of inapproximability gap, we show a stronger guarantee in the Yes Case. Specifically, we show that in the Yes case, there are at least $\frac{1}{2} - \varepsilon$ fraction of *vertices* on, say, the left side such that *all* the edges incident on them are satisfied by some assignment and also the instance is left-regular. This clearly implies the above theorem. Formally, the main theorem that we prove is (See Definition 7 for a formal definition of Unique Games):

▶ **Theorem 2.** *For every $\delta > 0$ there exists $L \in \mathbb{N}$ such that the following holds. Given an instance $G = (A, B, E, [L], \{\pi_e\}_{e \in E})$ of Unique Games, which is regular on the A side, it is NP-hard to distinguish between the following two cases:*

- *YES case: There exist $A' \subseteq A$ of size $(\frac{1}{2} - \delta)|A|$ and assignment that satisfies all the edges incident on A.*
- *NO case: Every assignment satisfies at most δ fraction of the edge constraints.*

We will denote by $\mathsf{val}(G)$ the maximum, over all assignments, fraction of edges satisfied and $\mathsf{sval}(G)$ to be the maximum, over all assignments, fraction of vertices in A such that all its edges are satisfied. Thus, the above theorem says that for every δ there exists a label set $[L]$ such that it is NP-hard to distinguish between the cases $\mathsf{sval}(G) \geqslant \frac{1}{2} - \delta$ and $\mathsf{val}(G) \leqslant \delta$.

1.1 $(\frac{1}{2} - \varepsilon)$-satisfiable UG vs. $(1 - \varepsilon)$-satisfiable UG

Let $\varepsilon > 0$ be a very small constant. In the $(1 - \varepsilon)$-satisfiable Unique Games instance, by simple averaging argument it follows that for any satisfying assignment $\sigma : A \cup B \to [L]$, there exists $A' \subseteq A$, $|A'| \geqslant (1 - \sqrt{\varepsilon})|A|$ such that for all $v \in A'$, at least $(1 - \sqrt{\varepsilon})$ fraction of edges of v are satisfied. Having such a large A' is crucial in many UG-reductions. For eg. a typical k-query inner verifier samples $v \in A$ and k neighbors of $u_1, u_2, \ldots, u_k$ of v u.a.r. Thus, with probability at least $(1 - \sqrt{\varepsilon})(1 - k\sqrt{\varepsilon}) \approx 1$ all the edges (u, v_i) are satisfied by any $(1 - \varepsilon)$ satisfying assignment σ.

In contrast to this, if we take $\frac{1}{2}$-satisfiable UG then the probability that all the edges (v, u_i) are satisfied is at most $\frac{1}{2^k}$ in the worst case. Therefore, in converting the known UG-hardness result to NP-hardness result using the NP-hardness of Unique Games with gap $(\frac{1}{2} - \varepsilon, \varepsilon)$, it is not always the case that we lose '*only half*' in the completeness case.

[1] A constraint $\pi : \Sigma_A \to \Sigma_B$ is called a d-to-1 projection constraint, if every $\beta \in \Sigma_B$ has exactly d pre-images.

Another important property of the Unique Games instance which was used in many reductions is that in the completeness case, there are 0.99 fraction of vertices on one side such that *all* the edges attached to them are satisfied i.e $\mathsf{sval}(G) \geqslant 1-\delta$ instead of $\mathsf{val}(G) \geqslant 1-\delta$. For eg., this property was crucial in the hardness of approximating independent sets in bounded degree graphs [3] and in many other reductions [7, 8].

As shown in [19], having completeness $\mathsf{val}(G) \geqslant 1-\delta$ for all sufficiently small $\delta > 0$ is equivalent to having completeness $\mathsf{sval}(G) \geqslant 1-\delta'$ for all sufficiently small $\delta' > 0$. It was crucial in the reduction that the $\mathsf{val}(G)$ is arbitrarily close to one for the equivalence to hold. We do not know a black-box way of showing the equivalence of $\mathsf{val}(G) = c$ and $\mathsf{sval}(G) = c$ for any $c < 1$. Thus, in order to prove Theorem 2, with a stronger completeness guarantee, we crucially exploit the structure of the game given by the known proofs of the 2-to-2 theorems.

1.2 Implications

Using Theorem 2, we show the following hardness results by going over the known reductions based on the Unique Games Conjecture.

Independent sets in degree d graphs

The first application is approximating maximum sized independent set in a degree d graph, where d is a large constant.

▶ **Theorem 3.** *It is NP-hard (under randomized reductions) to approximate independent sets in a degree d graph within a factor of $O\left(\frac{d}{\log^2 d}\right)$, where d is constant.*

This improves the NP-hardness of approximation within a factor $O\left(\frac{d}{\log^4 d}\right)$, as shown in Chan [9] as well as shows the tightness of the randomized polynomial time approximation algorithm given by Bansal *et al.* [6].

Max-Acyclic Subgraph

Given a directed graph $G(V, E)$, the Max-Acyclic Subgraph problem is to determine the maximum fraction of edges $E' \subseteq E$ such that removal of $E \setminus E'$ makes the graph acyclic (removes all the cycles). We can always make a graph acyclic by removing at most $\frac{1}{2}$ fraction of the edges and hence it gives a trivial $\frac{1}{2}$-approximation algorithm. Guruswami *et al.* [13] showed this is tight by showing that assuming the Unique Games Conjecture, it is NP-hard to approximate Max-Acyclic Subgraph within a factor of $\frac{1}{2} + \varepsilon$ for all $\varepsilon > 0$. In terms of NP-hardness, Austrin *et al.* [4] showed NP-hardness of approximating Max-Acyclic Subgraph within a ratio of $\frac{14}{15} + \varepsilon$, improving upon the previous bound of $\frac{65}{66} + \varepsilon$ by Newman [22]. Our next theorem shows an improved inapproximability of $\frac{2}{3} + \varepsilon$. One interesting feature of the hard instance is that it is hard to perform better than the trivial $\frac{1}{2}$-approximation on the instance.

▶ **Theorem 4.** *For all $\varepsilon > 0$, given a directed graph $G(V, E)$, it is NP-hard to approximate Max-Acyclic Subgraph problem within a factor of $\frac{2}{3} + \varepsilon$ for all $\varepsilon > 0$.*

We note that Theorem 1 along with the reduction from [13] imply NP-hardness of Max-Acyclic Subgraph problem within a factor of $\frac{4}{5} + \varepsilon$ (See Remark 35 for a proof sketch). Therefore, Theorem 4 improves upon this bound too.

- Select a $\ell - 1$ dimensional subspace L' u.a.r.
- Select a ℓ dimensional subspace L containing L' u.a.r.
- Check if $f(L)_{|L'} = h(L')$.

Figure 1 2-to-1 Test $\mathcal{T}_1$.

Predicates supporting balanced pairwise independent distributions

The next result is approximating Max-k-CSP(P) for a predicate $P : [q]^k \to \{0, 1\}$ where $P^{-1}(1)$ supports a balanced pairwise independent distribution. In [5], it was shown that assuming UGC, given a $(1 - \varepsilon)$-satisfiable instance of Max-k-CSP(P), it is hard to find an assignment that satisfies more than $\frac{|P^{-1}(1)|}{q^k} + \varepsilon$ fraction of the constraints for any constant $\varepsilon > 0$. Note that a random assignment satisfies $\frac{|P^{-1}(1)|}{q^k}$ fraction of the constraints in expectation and the theorem says that doing better than this even for almost satisfiable instance is UG-hard. If we instead use Theorem 2 as a starting point of the reduction, we get the following NP-hardness result.

▶ **Theorem 5.** *If a predicate $P : [q]^k \to \{0, 1\}$ supports a balanced pairwise independent distribution, then it is NP-hard to find a solution with value $\frac{P^{-1}(1)}{q^k} + \varepsilon$ if a given P-CSP instance is $\frac{1}{2} - \varepsilon$ satisfiable, for every $\varepsilon > 0$.*

Theorem 2 implies many more NP-hardness results in a straightforward way by going over the known reductions based on UGC, but we shall restrict ourselves to proving only the above three theorems. We only state the following important implication which follows from the result of Raghavendra [23] and our main theorem. We refer to [23] for the definition of (c, s) SDP integrality gap of a P-CSP instance.

▶ **Theorem 6** (Informal)**.** *For all $\varepsilon > 0$, if a P-CSP has (c, s) SDP integrality gap instance, then it is NP-hard to distinguish between $(\frac{c}{2} - \varepsilon)$-satisfiable instances from at most $(s + \varepsilon)$-satisfiable instances.*

The reduction actually gives a stronger result; Instead of completeness $(\frac{c}{2} - \varepsilon)$ one can get $(\frac{c}{2} + \frac{r}{2} - \varepsilon)$ where $r = \frac{|P^{-1}(1)|}{q^k}$ for a predicate $P : [q]^k \to \{0, 1\}$.

2 Overview

In this section, we give an overview of the proof of Theorem 2. The main idea which goes in proving Theorem 2 is very simple and we elaborate it next.

Let $V = \mathbb{F}_2^n$ and $\begin{bmatrix} V \\ \ell \end{bmatrix}$ denotes the set of all ℓ dimensional subspaces of V. Consider the following Grassmann 2-to-1 test $\mathcal{T}_1$ for functions $f : \begin{bmatrix} V \\ \ell \end{bmatrix} \to \mathbb{F}_2^\ell$ and $h : \begin{bmatrix} V \\ \ell-1 \end{bmatrix} \to \mathbb{F}_2^{\ell-1}$, where for a subspace L (L'), $f(L)$ ($h(L')$) represents a linear function on the subspace, by fixing an arbitrarily chosen basis of L (L').

From the test it is clear that for every pair (L, L') such that $L' \subseteq L$, for every linear function β on L', there are linear functions α_1, α_2 on L such that the test passes for any pair (α_i, β). This gives the 2-to-1 type constraints.

One way to convert a 2-to-1 test to a unique test is by choosing a random $i \in \{1, 2\}$ for every pair (L, L') such that $L' \subseteq L$ and for every linear function β on L', and adding the accepting pair (α_i, β) where $\{(\alpha_1, \alpha_2), \beta\}$ are the original accepting assignments. This does

- Select a $\ell - 1$ dimensional subspace L' u.a.r.
- Select a ℓ dimensional subspace L containing L' u.a.r. and $x \in L \setminus L'$ u.a.r.
- Check if $f(L,x)_{|L'} = h(L')$.

Figure 2 Unique Test $\mathcal{T}_2$.

give a unique test and if f and h are restrictions of a global linear function to the subspaces, then with high probability the test passes with probability $\approx \frac{1}{2}$. One drawback of this test is that, if we consider a bipartite graph on $\begin{bmatrix} V \\ \ell \end{bmatrix} \times \begin{bmatrix} V \\ \ell-1 \end{bmatrix}$ where two subspaces L, L' are connected iff $L' \subseteq L$, then for any global linear function we can only argue that half the edges are *satisfied* in the sense of the unique test. Note that the uniform distribution on the edges of this bipartite graph is the same as the test distribution $\mathcal{T}_1$. Hence, the similar guarantee of satisfying around half the edges stays in the final Unique Games instance created from the works of [16, 10, 11, 17] and hence falls short of proving Theorem 2.

Now we convert it into a Unique Test $\mathcal{T}_2$ with a guarantee that for around $\frac{1}{2}$ fraction of the vertices, *all* the edges incident on them are satisfied if the assignments f and h are restrictions of a global linear function. Towards this, we modify the domain of f. We consider two functions $f : \begin{bmatrix} V \\ \ell \end{bmatrix} \times 2^{[\ell]} \to \mathbb{F}_2^{\ell}$ and $h : \begin{bmatrix} V \\ \ell-1 \end{bmatrix} \to \mathbb{F}_2^{\ell-1}$. We fix an arbitrary one-to-one correspondence between the elements of ℓ dimension subspace and $2^{[\ell]}$. Thus, we can now interpret f as defined on tuples (L, x) where $x \in L$. We consider the assignments $f(L, x)$ and $h(L')$ as linear functions on spaces L and L' respectively. Consider the following bipartite graph $(\begin{bmatrix} V \\ \ell \end{bmatrix} \times 2^{[\ell]}, \begin{bmatrix} V \\ \ell-1 \end{bmatrix}, E)$ where (L, x) is connected to L' iff $x \notin L'$ and $L' \subseteq L$. The test distribution which we will define next will be uniform on the edges of this graph.

We now put permutation constraint on the edges of the graph. For each vertex (L, x) we select $b_{L,x} \in \{0, 1\}$ u.a.r. For an edge $e \in E$ between (L, x) and L' we set the following unique constraint: Extend the linear function given by h on L' to a linear function $\tilde{h}$ on $\mathtt{span}\{L', x\}$ by setting $\tilde{h}(x) = b_{L,x}$. The accepting labels for an edge e are $f(L, x)$ and $h(L')$ such that $\tilde{h}(\mathtt{span}\{L', x\})_{|L'}$ and $f(L, x)|_{L'}$ are identical. Note that once $b_{L,x}$ is chosen, for every label $f(L, x)$ there is a unique label to its neighbor L' which satisfies the constraint and also vice-versa.

Suppose (f, h) are restrictions of a fixed global linear function $g : V \to \mathbb{F}_2$ to the respective subspaces. In this case, if $b_{L,x} \in \{0, 1\}$ is such that $g(x) = b_{L,x}$, then the assignments (f, h) satisfy all the edges incident on (L, x). This is because for any edge between (L, x) and L', we have $\tilde{h}(\mathtt{span}\{L', x\})|_{L'} = f(L, x)|_{L'} = g|_{L'}$. Since the event $g(x) = b_{L,x}$ happens with probability $\frac{1}{2}$, we get that with high probability for at least $(\frac{1}{2} - \varepsilon)$ fraction of the vertices on the left, all the edges incident on it are satisfied by the assignment (f, h) for any constant $\varepsilon > 0$.

3 Preliminaries

We start by defining the Unique Games.

▶ **Definition 7** (Unique Games). *An instance $G = (A, B, E, [L], \{\pi_e\}_{e \in E})$ of the Unique Games constraint satisfaction problem consists of a bipartite graph (A, B, E), a set of alphabets $[L]$ and a permutation map $\pi_e : [L] \to [L]$ for every edge $e \in E$. Given a labeling $\ell : A \cup B \to [L]$, an edge $e = (u, v)$ is said to be satisfied by ℓ if $\pi_e(\ell(v)) = \ell(u)$.*

G is said to be at most δ-satisfiable *if every labeling satisfies at most a δ fraction of the edges.*

We will define the following two quantities related to the satisfiability of the Unique Games instance.

$$\mathsf{val}(G) := \max_{\sigma: A\cup B\to[L]} \{\text{fraction of edges in } G \text{ satisifed by } \sigma\}.$$

$$\mathsf{sval}(G) := \max_{\sigma: A\cup B\to[L]} \left\{ \frac{|A'|}{|A|} \mid \forall e(u,v) \text{ s.t. } u\in A, e \text{ is satisfied by } \sigma \right\}.$$

The following is a conjecture by Khot [14] which has been used to prove many *tight* inapproximability results.

▶ **Conjecture 8** (Unique Games Conjecture [14]). *For every sufficiently small $\delta > 0$ there exists $L \in \mathbb{N}$ such that given a an instance $\mathcal{G} = (A, B, E, [L], \{\pi_e\}_{e\in E})$ of Unique Games it is NP-hard to distinguish between the following two cases:*

- *YES case: $\mathsf{val}(G) \geqslant 1-\delta$.*
- *NO case: $\mathsf{val}(G) \leqslant \delta$.*

For a linear subspace $L \subseteq \mathbb{F}_2^n$, the dimension of L is denoted by $\mathtt{dim}(L)$. For two subspaces $L_1, L_2 \subseteq \mathbb{F}_2^n$, we will use $\mathtt{span}(L_1, L_2)$ to denote the subspace $\{x_1 + x_2 \mid x_1 \in L_1, x_2 \in L_2\}$. We will sometimes abuse the notation and write $\mathtt{span}(x, L)$,where $x \in \mathbb{F}_2^n$, to denote $\mathtt{span}(\{0, x\}, L)$. For subspaces L_1, L_2 such that $L_1 \cap L_2 = \{0\}$, define $L_1 \oplus L_2 := \mathtt{span}(L_1, L_2)$.

For $0 < \ell < n$, let $Gr(\mathbb{F}_2^n, \ell)$ be the set of all ℓ dimensional subspaces of $\mathbb{F}_2^n$. Similarly, for a subspace L of $\mathbb{F}_2^n$ such that $\mathtt{dim}(L) > \ell$, let $Gr(L, \ell)$ be the set of all ℓ dimensional subspaces of $\mathbb{F}_2^n$ *contained* in L.

4 The Reduction

In this section, we go over the reduction in [11] from a gap 3LIN instance to a 2-to-1 Label Cover instance and then show how to reduce it to a Unique Games instance in Section 4.4. We retain most of the notations from [11].

4.1 Outer Game

The starting point of the reduction is the following problem:

▶ **Definition 9** (Reg-3Lin). *The instance (X, Eq) of* Reg-3Lin *consists if variables $X = \{x_1, x_2, \ldots, x_n\}$ taking values in $\mathbb{F}_2$ and $\mathbb{F}_2$ linear constraints $e_1, e_2, \ldots, e_m$, where each e_i is a linear constraint on 3 variables. The instance is regular in the following ways: every equation consists of 3 distinct variables, every variable x_i appears in exactly 5 constraints and every two distinct constraints share at most one variable.*

An instance (X, Eq) is said to be t-satisfiable if there exists an assignment to X which satisfies t fraction of the constraints. We have the following theorem implied by the PCP theorem of [1, 2, 12].

▶ **Theorem 10.** *There exists an absolute constant $s < 1$ such that for every constant $\varepsilon > 0$ it is NP-hard to distinguish between the cases when the instance is at least $(1-\varepsilon)$ satisfiable vs. at most s satisfiable.*

We now define an outer 2-prover 1-round game, parameterized by $k, q \in \mathbb{Z}^+$ and $\beta \in (0, 1)$, which will be the starting point of our reduction. The verifier selects k constraints $e_1, e_2, \ldots, e_k$ from the instance (X, Eq) uniformly at random with repetition. If e_i and e_j share a variable

for some $i \neq j$ then accept. otherwise, Let $x_{i,1}, x_{i,2}, x_{i,3}$ be the variables in constraint e_i. Let $X_1 = \cup_{i=1}^{k} \{x_{i,1}, x_{i,2}, x_{i,3}\}$. The verifier then selects a subset X_2 of X_1 as follows: for each $i \in [k]$, with probability $(1-\beta)$ add $x_{i,1}, x_{i,2}, x_{i,3}$ to X_2 and with probability β, select a variable from $\{x_{i,1}, x_{i,2}, x_{i,3}\}$ uniformly at random and add it to X_2.

On top of this, the verifier selects q pair of *advice* strings (s_j, s_j^*) where $s_j \in \{0,1\}^{X_2}$, and $s_j^* \in \{0,1\}^{X_1}$ for $1 \leq j \leq q$ as follows : For each $j \in [q]$, select $s_j \in \{0,1\}^{X_2}$ uniformly at random. The string s_j can be though as assigning bits to each of the variables from X_2. The string $s_j^* \in \{0,1\}^{3k}$ is deterministically selected such that its projection on X_2 is same as s_j and the rest of the coordinates are filled with 0.

The verifier sends $(X_1, s_1^*, s_2^*, \ldots, s_q^*)$ to prover 1 and $(X_2, s_1, s_2, \ldots, s_q)$ to prover 2. The verifier expects an assignment to variables in X_i from prover i. The verifier accepts if and only if the assignment to X_1 given by prover 1 satisfies all the equations $e_1, e_2, \ldots, e_k$ and the assignment X_2 given by prover 2 is consistent with the answer of prover 1.

Completeness: It is easy to see the completeness case. If the instance (X, Eq) is $(1-\varepsilon)$ satisfiable then there is a provers' strategy which makes the verifier accepts with probability at least $(1-k\epsilon)$. The strategy is to use a fixed $(1-\varepsilon)$-satisfiable assignment and answer according to it. In this case, with probability at least $(1-k\epsilon)$, the verifier chooses k constraints which are all satisfied by the fixed assignment and hence the verifier will accept provers' answers.

Soundness: Consider the case when the instance (X, Eq) is at most s-satisfiable for $s < 1$ from Theorem 10. If the provers were given only X_1 and X_2 without the advice strings, then the parallel repetition theorem of Raz [24] directly implies that for any provers' strategy, they can make the verifier accept with probability at most $2^{-\Omega(\beta k)}$. It turns out that a few advice strings will not give provers any significant advantage. This is formalized in the following theorem.

▶ **Theorem 11** ([16]). *If the* Reg-3Lin *instance* (X, Eq) *is at most* $s < 1$ *(from Theorem 10) satisfiable then there is no strategy with which the provers can make the verifier accept with probability greater than* $2^{-\Omega(\beta k/2^q)}$.

▶ Remark 12. The importance of advice strings will come later in the proof of soundness. Specifically, the proof of Theorem 23 (from [11] which we use as a black-box) crucially uses the advice strings given to the provers.

To prove our main theorem, the reduction is carried out in three steps:

$$
\begin{array}{ll}
\text{Outer Game} & \\
\quad \downarrow \text{ [11]} & \\
G_{\texttt{unfolded}}(A, B, E, \Pi, \Sigma_A, \Sigma_B) & \text{(unfolded 2-to-1 Game)} \\
\quad \downarrow \text{ [11]} & \\
G_{\texttt{folded}}(\tilde{A}, B, \tilde{E}, \tilde{\Pi}, \Sigma_A, \Sigma_B) & \text{(folded 2-to-1 Game)} \\
\quad \downarrow \text{(This work)} & \\
\mathsf{UG}_{\texttt{folded}}(\widehat{A}, B, \widehat{E}, \widehat{\Pi}, \Sigma) & \text{(Unique Game)}
\end{array}
$$

The first two steps are explained in the next two subsections. These follow from [11]. The main contribution of our work is the last step which is given in Section 4.4.

4.2 Unfolded 2-to-1 Game

In this section we reduce Reg-3Lin to an instance of 2-to-1 Label cover instance $G_{\texttt{unfolded}} = (A, B, E, \Pi, \Sigma_A, \Sigma_B)$.

A set of k equations $(e_1, e_2, \ldots, e_k)$ is *legitimate* if the support of equations are pairwise disjoint and for every two different equations e_i and e_j and for any $x \in e_i$ and $y \in e_j$, the pair $\{x, y\}$ does not appear in any equation in (X, Eq). Let $\mathcal{U}$ be the set of all legitimate tuples of k equations. For $U \in \mathcal{U}$, Let $X_U \subseteq \mathbb{F}_2^n$ be the subspace with support in $U \subseteq [n]$. For an equation $e = (i, j, k) \in U$, let x_e be a vector in X_U where $x_i = x_j = x_k = 1$ and rest of the coordinates are 0. Denote by $b_e \in \mathbb{F}_2$ the RHS of the equation e. Let H_U be the span of $\{x_e : e \in U\}$. Finally, Let $\mathcal{V}$ be the collection of all sets of variables upto size $3k$..

Vertices (A, B)

Let $\ell \ll k$ which we will set later. The vertex set of the game $G_{\texttt{unfolded}}$ is defined as follows:

$$A = \{(U, L) \mid U \in \mathcal{U}, L \in Gr(X_U, \ell), L \cap H_U = \{0\}\}.$$

$$B = \{(V, L') \mid V \in \mathcal{V}, L' \in Gr(X_V, \ell - 1)\}.$$

Edges E

The distribution on edges are defined by the following process: Choose X_1 and X_2 as per the distribution given in the outer verifier conditioned on $X_1 \in \mathcal{U}$. Let $U = X_1$ and $V = X_2$. Choose a random subspace $L' \in Gr(X_V, \ell - 1)$ and a random $L \in Gr(X_U, \ell)$ such that $L' \subseteq L$. Output $\{(U, L), (V, L')\} \in (A, B)$.

Labels (Σ_A, Σ_B)

The label set $\Sigma_A = \mathbb{F}_2^\ell$ and the label set $\Sigma_B = \mathbb{F}_2^{\ell-1}$. A labeling $\sigma \in \Sigma_A$ to (U, L) can be thought of as a linear function $\sigma : L \to \mathbb{F}_2$. Similarly the label $\sigma' \in \Sigma_B$ to a vertex (V, L') is though of as a linear function $\sigma' : L' \to \mathbb{F}_2$. This can be done by fixing arbitrary basis of the respective spaces.

4.3 Folded 2-to-1 Game

For every assignment to the 3LIN instance, there are many vertices in the graph $G_{\texttt{unfolded}}$ which get the same label according to strategy of labeling the vertices in $G_{\texttt{unfolded}}$ with respect to the assignment. So we might as well enforce this constraint on the variables in $G_{\texttt{unfolded}}$. This is acheived by *folding*. In this section, we convert $G_{\texttt{unfolded}}$ to the following Game $G_{\texttt{folded}} = (\tilde{A}, B, \tilde{E}, \tilde{\Pi}, \Sigma_A, \Sigma_B)$.

Vertices $(\tilde{A}, B)$

Consider the following grouping of the vertices from A

$$\mathcal{C}(U_0, L_0) = \{(U, L) \in A \mid L_0 \oplus H_U \oplus H_{U_0} = L \oplus H_U \oplus H_{U_0}\}.$$

The following Lemma 13 says that $\mathcal{C}$ is indeed an equivalence class. We define the vertex set $\tilde{A}$ as follows:

$$\tilde{A} = \{\mathcal{C}(U, L) \mid (U, L) \in A\}.$$

In other words, there is a vertex for every equivalence class in $\tilde{A}$.

▶ **Lemma 13** ([11]). *$\mathcal{C}$ is an equivalence class: there exists a ℓ dimension subspace $R_{\mathcal{C}}$ such that for all $(U, L) \in \mathcal{C}$,*

$$H_U \oplus L = R_{\mathcal{C}} \oplus H_U.$$

Edges $\tilde{E}$

Sample (U, L) (V, L') with respect to E. Output $\mathcal{C}(U, L)$, (V, L').

Labels (Σ_A, Σ_B)

The label set $\Sigma_A = \mathbb{F}_2^\ell$, a label σ to $\mathcal{C}$ can be thought of as a linear function $\sigma : R_{\mathcal{C}} \to \mathbb{F}_2$. As before, the label $\sigma' \in \Sigma_B$ to a vertex (V, L') is though of as a linear function $\sigma' : L' \to \mathbb{F}_2$.

In order to define the constraints on the edges, we need the following definitions:

▶ **Definition 14.** *For a space $H_U \oplus L$ such that $L \cap H_U = \{0\}$ and a linear function $\sigma : L \to \mathbb{F}_2$, the extension of σ, respecting side conditions, to the whole space $H_U \oplus L$ is a linear function $\beta : H_U \oplus L \to \mathbb{F}_2$ such that for all $e \in U$, $\beta(x_e) = b_e$ and $\beta|_L = \sigma$.*

Note that there is one to one mapping from a linear function on L and its extension as all the equations in U are disjoint and hence $\{x_e \mid e \in U\}$ form a basis of the space H_U.

▶ **Definition 15.** *Consider a label σ to a vertex $\mathcal{C}$ which is a linear function on $R_{\mathcal{C}}$. The unfolding of it to the elements of the $\mathcal{C}$ is given as follows: For $(U, L) \in \mathcal{C}$, define a linear function $\tilde{\sigma}_U : H_U \oplus L \to \mathbb{F}_2$ such that it is equal to the extension of σ to $H_U \oplus R_{\mathcal{C}}$ respecting side conditions.*

The spaces $H_U \oplus L$ and $H_U \oplus R_{\mathcal{C}}$ are the same and hence the above definition makes sense. We are now ready to define the constraints.

Constraints $\tilde{\Pi}$

Consider linear functions $\sigma : R_{\mathcal{C}} \to \mathbb{F}_2$ and $\sigma' : L' \to \mathbb{F}_2$. A pair (σ, σ') satisfies the edge $(\mathcal{C}, (V, L')) \in \tilde{E}$, if for every $(U, L) \in \mathcal{C}$ such that $((U, L), (V, L')) \in E$, the unfolding $\tilde{\sigma}_U|_{L'} = \sigma'$.

We have the following completeness and soundness guarantee of the reduction from [11].

▶ **Lemma 16** (Completeness). *If the* Reg-3Lin *instance (X, Eq) is $(1 - \varepsilon)$ satisfiable then there exists $\tilde{A}' \subseteq \tilde{A}$, $|\tilde{A}'| \geqslant (1 - k\epsilon)|\tilde{A}|$ and a labeling to the 2-to-1 Label Cover instance $G_{\texttt{folded}}$ such that* all *the edges incident on $\tilde{A}'$ are satisfied.*

▶ **Lemma 17** (Soundness). *For all $\delta > 0$, there exists $q, k \geq 1$ and $\beta \in (0, 1)$, such that if the* Reg-3Lin *instance (X, Eq) is at most s satisfiable (where s is from Theorem 10) then every labeling to $G_{\texttt{folded}}$ satisfies at most δ fraction of the edges.*

4.4 Reduction to Unique Games

In this section, we convert $G_{\texttt{folded}}$ Label Cover instance to a Unique Games instance with the stronger completeness guarantee that we are after. We will reduce an instance $G_{\texttt{folded}} = (\tilde{A}, B, \tilde{E}, \tilde{\Pi}, \Sigma_A, \Sigma_B)$ to an instance of Unique Game $\mathsf{UG}_{\texttt{folded}} = (\widehat{A}, B, \widehat{E}, \widehat{\Pi}, \Sigma)$.

Vertices $(\widehat{A}, B)$

We will split each vertex $\mathcal{C} \in \tilde{A}$ into many copies. Fix an ℓ dimensional subspace $R_{\mathcal{C}}$ given by Lemma 13. For every $x \in R_{\mathcal{C}}$ and $b \in \{0,1\}$ we add a copy $C_{x,b}$ to $\widehat{A}$.

$$\widehat{A} = \bigcup_{\mathcal{C} \in \tilde{A}} \{\mathcal{C}_{x,b} \mid x \in R_{\mathcal{C}}, b \in \{0,1\}\}.$$

Edges $\widehat{E}$

The distribution on the edge set $\widehat{E}$ is as follows: We first pick $((U,L),(V,L'))$ according to the distribution E. Let $(U,L) \in \mathcal{C}$. We then select $y \in (H_U \oplus L) \setminus (H_U \oplus L')$ and $b \in \{0,1\}$ uniformly at random. Note that $\mathtt{dim}(\mathtt{span}\{y, H_U\} \cap R_{\mathcal{C}}) = 1$ since $y \notin H_U$. Let $x \in \mathtt{span}\{y, H_U\} \cap R_{\mathcal{C}}$ be the non-zero vector. Output $(\mathcal{C}_{x,b}, (V,L'))$.

▷ Claim 18. x is distributed uniformly in $R_{\mathcal{C}} \setminus (H_U \oplus L')$ conditioned on (U,V,L,L').

Proof. We first claim that $x \in R_{\mathcal{C}} \setminus (H_U \oplus L')$ by showing $x \notin H_U \oplus L'$. Suppose not, then we can write $x = h + x'$ where $h \in H_U$ and $x' \in L'$. We also know that $x \in \mathtt{span}\{y, H_U\}$ and thus x can be written as $x = \tilde{h} + y$ where $\tilde{h} \in H_U$. This implies $h + x' = \tilde{h} + y$. In other words, $y = h + \tilde{h} + x' \in H_U \oplus L'$, a contradiction.

Since each $y \in (H_U \oplus L) \setminus (H_U \oplus L')$ gives an unique non-zero $x \in \mathtt{span}\{y, H_U\} \cap R_{\mathcal{C}}$, we will show that the number of $y \in (H_U \oplus L) \setminus (H_U \oplus L')$ which gives a fixed x is same for all $x \in R_{\mathcal{C}} \setminus (H_U \oplus L')$ and this will prove the claim.

Fix any $\tilde{x} \in R_{\mathcal{C}} \setminus (H_U \oplus L')$. We now claim that the set of all $y \in (H_U \oplus L) \setminus (H_U \oplus L')$ that gives $\tilde{x}$ is $\mathtt{span}\{\tilde{x}, H_U\} \setminus H_U$. Clearly, for any $y \notin \mathtt{span}\{\tilde{x}, H_U\} \setminus H_U$, $\tilde{x} \notin \mathtt{span}\{y, H_U\}$ and also for every $y \in \mathtt{span}\{\tilde{x}, H_U\} \setminus H_U$, $\tilde{x} \in \mathtt{span}\{y, H_U\}$. Thus, it remains to show that $\mathtt{span}\{x, H_U\} \setminus H_U \subseteq (H_U \oplus L) \setminus (H_U \oplus L')$ for all $x \in R_{\mathcal{C}} \setminus (H_U \oplus L')$.

To prove the inclusion, suppose for contradiction $\mathtt{span}\{x, H_U\} \cap (H_U \oplus L') \neq \emptyset$. This means $x + h = \tilde{h} + v'$ for some $h, \tilde{h} \in H_U$ and $v' \in L'$. This implies $x = h + \tilde{h} + v' \in H_U \oplus L'$ contradicting $x \in R_{\mathcal{C}} \setminus (H_U \oplus L')$. ◁

Labels Σ

The label set $\Sigma = \mathbb{F}_2^{\ell-1}$, a label σ to $\mathcal{C}_{x,b}$ can be thought of as a linear function $\sigma : R_{\mathcal{C}} \to \mathbb{F}_2$ such that $\sigma(x) = b$. It is easy to see that there is a one-to-one correspondence between a label σ and a linear function $\tilde{\sigma}$ on $R_{\mathcal{C}}$. Similar to the previous case of $G_{\mathtt{folded}}$, a label from $\Sigma (= \Sigma_B)$ to a vertex (V,L') in B is interpreted as a linear function $\sigma' : L' \to \mathbb{F}_2$.

We define an analogous unfolding of label to vertices in $\widehat{A}$ to the elements of the corresponding equivalence class. Since the label sets are different, for a label σ to $\mathcal{C}_{x,b}$ (thought of as a linear function on $R_{\mathcal{C}}$ respecting $\sigma(x) = b$) we use the notation $\widehat{\sigma}_U$ to denote its unfolding to $(U,L) \in \mathcal{C}_{x,b}$.

1-to-1 Constraints $\widehat{\Pi}$

Finally the constraint $\pi_e : \Sigma \to \Sigma$ between the endpoints of an edge $e = (\mathcal{C}_{x,b}, (V,L'))$ is given as follows: Consider linear functions $\sigma : R_{\mathcal{C}} \to \mathbb{F}_2$ respecting $\sigma(x) = b$ and $\sigma' : L' \to \mathbb{F}_2$. A pair $(\sigma, \sigma') \in \pi_e$ if for every $(U,L) \in \mathcal{C}$ such that $((U,L),(V,L')) \in E$ and $\mathtt{span}\{x, H_U\} \cap L' = \{0\}$, the unfolding $\tilde{\sigma}_U$ satisfies $\tilde{\sigma}_U|_{L'} = \sigma'$.

To see that every σ' has a unique preimage, for any linear function $\sigma' : L' \to \mathbb{F}_2$, there is a unique linear function $\sigma : R_{\mathcal{C}} \to \mathbb{F}_2$ such that $\sigma(x) = b$ satisfying the above conditions. This is because of the following claim.

▷ Claim 19. Any basis for L' along with x and $\{x_e : e \in U\}$ form a basis for $H_U \oplus R_{\mathcal{C}}$ for every $(U, L) \in \mathcal{C}$.

Proof. Let us unwrap the conditions for putting an edge between (V, L') and $\mathcal{C}_{x,b}$. One necessary condition is that $(\mathcal{C}, (V, L'))$ should be an edge in $\tilde{E}$. By the definition of $\tilde{E}$, there exists $(U, L) \in \mathcal{C}$ such that $L' \subseteq L$. Recall, x is such that there exists $y \in (H_U \oplus L) \setminus (H_U \oplus L')$ such that $\mathtt{dim}(\mathtt{span}\{y, H_U\} \cap R_{\mathcal{C}}) = 1$ and $x \in \mathtt{span}(y, H_U) \cap R_{\mathcal{C}}$. Therefore $x \in (H_U \oplus L) \setminus (H_U \oplus L')$ and hence $\mathtt{dim}(\mathtt{span}\{x, H_U \oplus L'\}) = k + \ell$ (as $H_U \cap L' = \{0\}$). This implies that any basis of L', basis $\{x_e : e \in U\}$ of H_U and x span $H_U \oplus L$. Since by Lemma 13 the space $H_U \oplus L$ is same as the space $H_U \oplus R_{\mathcal{C}}$, the claim follows. ◁

We now show the completeness and soundness of the Unique Games instance:

▶ **Theorem 20** ([11]). *If the* Reg-3Lin *instance is* $(1 - \varepsilon)$*-satisfiable, then there exists* $\tilde{A}' \subseteq \tilde{A}$, $|\tilde{A}'| \geqslant (1 - k\epsilon)|\tilde{A}|$ *and a labeling to the* 2*-to-*1 *Label Cover instance* $G_{\mathtt{folded}}$ *such that* all *the edges incident on* $\tilde{A}'$ *are satisfied.*

▶ **Lemma 21** (Completeness). *For all* $\varepsilon > 0$*, if there exists* $\tilde{A}' \subseteq \tilde{A}$, $|\tilde{A}'| \geqslant (1 - k\epsilon)|\tilde{A}|$ *and a labeling to the* 2*-to-*1 *Label Cover instance* $G_{\mathtt{folded}}$ *such that* all *the edges incident on* $\tilde{A}'$ *are satisfied then there exists* $\widehat{A'} \subseteq \widehat{A}$, $|\widehat{A'}| \geqslant (\frac{1-k\epsilon}{2})|\widehat{A}|$ *and a labeling to Unique Games instance* $\mathsf{UG}_{\mathtt{folded}}$ *such that* all *the edges incident on* $\widehat{A'}$ *are satisfied.*

Proof. Fix a labeling $(\tilde{\mathcal{A}}, \tilde{\mathcal{B}})$ to $G_{\mathtt{folded}}$ where $\tilde{\mathcal{A}} : \tilde{A} \to \Sigma_A$ and $\tilde{\mathcal{B}} : B \to \Sigma_B$ which satisfies all the edges incident on $(1 - k\epsilon)$ fraction of the vertices in $\tilde{A}$. We will construct a labeling $(\widehat{\mathcal{A}}, \widehat{\mathcal{B}})$ to the instance $\mathsf{UG}_{\mathtt{folded}}$, where $\widehat{\mathcal{A}} : \widehat{A} \to \Sigma$ and $\widehat{\mathcal{B}} : B \to \Sigma$ which will satisfy all the edges adjacent to at least $\frac{(1-k\epsilon)}{2}$ fraction of vertices $\widehat{A}$ in $\mathsf{UG}_{\mathtt{folded}}$.

We will set $\widehat{\mathcal{B}} = \tilde{\mathcal{B}}$. Now to assign a label to $\mathcal{C}_{x,b} \in \widehat{A}$, we look at the labeling $\sigma := \tilde{\mathcal{A}}(\mathcal{C}) \in \mathbb{F}_2^\ell$ as a linear function $\sigma : R_{\mathcal{C}} \to \mathbb{F}_2$. If $\sigma(x) = b$, we set $\widehat{\mathcal{A}}(\mathcal{C}_{x,b})$ to be the same linear function $\sigma : R_{\mathcal{C}} \to \mathbb{F}_2$ respecting $\sigma(x) = b$. Otherwise, we set $\widehat{\mathcal{A}}(\mathcal{C}_{x,b}) = \perp$. It is obvious that exactly half the vertices in $\widehat{A}$ got assigned a label in Σ.

▷ Claim 22. If the label $\tilde{\mathcal{A}}(\mathcal{C})$ to $\mathcal{C}$ satisfies all the edges incident on it, then the label $\widehat{\mathcal{A}}(\mathcal{C}_{x,b})$ satisfies all the edges incident on $\mathcal{C}_{x,b}$, unless $\widehat{\mathcal{A}}(\mathcal{C}_{x,b}) = \perp$.

Proof. For convenience let $\sigma = \tilde{\mathcal{A}}(\mathcal{C})$. If we let $\Gamma(\mathcal{C}) \subseteq B$ to be the neighbors of $\mathcal{C}$ in $G_{\mathtt{folded}}$, then the set of neighbors of $\mathcal{C}_{x,b}$ is a subset of $\Gamma(\mathcal{C})$. Furthermore if (V, L') is connected to $\mathcal{C}_{x,b}$ in $\mathsf{UG}_{\mathtt{folded}}$ then $x \notin L'$ and $x \in R_{\mathcal{C}}$. The condition that the edge $(\mathcal{C}, (V, L'))$ is satisfied by $\tilde{\mathcal{A}}$ means that for all $(U, L) \in \mathcal{C}$ such that $L' \subseteq L$, the unfolding of σ satisfies $\tilde{\sigma}_U|_{L'} = \tilde{B}((V, L'))$. Since the unfolding of the label $\widehat{\mathcal{A}}(\mathcal{C}_{x,b})$ to $\mathcal{C}_{x,b}$ gives the same linear function $\tilde{\sigma}$, it follows that $\tilde{\sigma}_U|_{L'} = \widehat{B}((V, L'))$ for every $(U, L) \in \mathcal{C}$ and every $(V, L') \in \Gamma(\mathcal{C})$ such that $L' \subseteq L$. Therefore $\widehat{\mathcal{A}}$ satisfies all the edges incident on $\mathcal{C}_{x,b}$. ◁

Let $\tilde{A}' \subseteq \tilde{A}$ be the set of vertices such that all the edges incident on them are satisfied by labeling $(\tilde{\mathcal{A}}, \tilde{\mathcal{B}})$. By assumption $|\tilde{A}'| \geqslant (1 - k\epsilon)|\tilde{A}|$. Consider the subset $\widehat{A'} \subseteq \widehat{A}$

$$\widehat{A'} = \{\mathcal{C}_{x,b} \mid \widehat{\mathcal{A}}(\mathcal{C}_{x,b}) \neq \perp, \mathcal{C} \in \tilde{A}'\}.$$

Now, $|\widehat{A'}| \geqslant \frac{1-k\epsilon}{2}|\widehat{A}|$ and from the above claim, all the edges incident on $\widehat{A'}$ are satisfied by the labeling $(\widehat{\mathcal{A}}, \widehat{\mathcal{B}})$. ◀

4.5 Soundness

Define $\texttt{agreement}(F_U)$ for $F_U : \{L \oplus H_U : L \in Gr(X_U, \ell), L \cap H_U = \{0\}\} \to \mathbb{F}_2^{\ell+3k}$, respecting side conditions, as the probability of the following event:

- Select a $\ell - 1$ dimension subspace $L' \in X_U$ u.a.r.
  - Select a ℓ dimension subspaces L_1 and L_2 containing L' u.a.r.
  - Check if $F_U[L_1 \oplus H_U]|_{L'} = F_U[L_1 \oplus H_U]|_{L'}$.

The main technical theorem which was conjectured in [11] and proved in [17] is that if $\texttt{agreement}(F_U)$ is a constant bounded away from 0, then there is a global linear function $g : X_U \to \{0,1\}$ respecting the side conditions and a special (not too small) subset S of $\{L \oplus H_U : L \in Gr(X_U, \ell), L \cap H_U = \{0\}\}$ such that for a constant fraction of elements in S, F_U agrees with g. We will not need the details of this theorem. Instead, we state the main soundness lemma from [11] which crucially used the aforementioned structural theorem and also the advice strings as mentioned in Remark 12.

▶ **Theorem 23** ([11]). *For every constant $\delta > 0$, there exist large enough $\ell \ll k$, $q \in \mathbb{Z}^+$ and $\beta \in (0,1)$ such that if there is an unfolded assignment $\mathcal{A} : A \to \Sigma_A$ to $G_{\texttt{unfolded}}$ such that for at least δ fraction of U, $\texttt{agreement}(F_U) \geqslant \delta$, then there exists a provers' strategy which makes the outer verifier accepts with probability at least p_δ, where p_δ is independent of k.*

Armed with this theorem, we are ready to prove the soundness of the Unique Games instance $\mathsf{UG}_{\texttt{folded}}$.

▶ **Lemma 24** (Soundness). *Let $\delta > 0$ and fix $q \in \mathbb{Z}^+$ and $\beta \in (0,1)$ and $\ell \ll k$ as in Theorem 23. If $\mathsf{UG}_{\texttt{folded}}$ is δ satisfiable then there exists a provers strategy which makes the outer verifier accepts with probability at least $p_{\frac{\delta^4}{2^{16}}}$.*

Proof. Fix any δ-satisfiable assignment $(\widehat{\mathcal{A}}, \widehat{\mathcal{B}})$, $\widehat{\mathcal{A}} : \widehat{A} \to \Sigma$, $\widehat{\mathcal{B}} : \widehat{B} \to \Sigma$ to the Unique Games instance $\mathsf{UG}_{\texttt{folded}}$. We first get a randomized labeling $(\tilde{\mathcal{A}}, \tilde{\mathcal{B}})$ to $G_{\texttt{folded}}$ where $\tilde{\mathcal{A}} : \tilde{A} \to \Sigma_A$ and $\tilde{\mathcal{B}} : B \to \Sigma_B$ as follows: We will keep $\tilde{\mathcal{B}} = \widehat{\mathcal{B}}$. For every $\mathcal{C} \in \tilde{A}$, we pick a random $x \in R_{\mathcal{C}}$ and $b \in \{0,1\}$ and set $\tilde{\mathcal{A}}(\mathcal{C}) = \widehat{\mathcal{A}}(\mathcal{C}_{x,b})$. We now unfold the assignment $\tilde{\mathcal{A}}$ to $\mathcal{A}$. Define $F_U[L] = \mathcal{A}(U, L)$ for every $L \in Gr(X_U, \ell)$.

Let $p(U)$ denote the probability that an edge in $\mathsf{UG}_{\texttt{folded}}$ is satisfied conditioned on U. Consider U such that $p(U) \geqslant \frac{\delta}{2}$. By an averaging argument, there are at least $\frac{\delta}{2}$ fraction of U such that $p(U) \geqslant \frac{\delta}{2}$.

▷ Claim 25. $\mathbf{E}_{F_U}[\texttt{agreement}(F_U)] \geqslant \frac{p(U)^4}{2^{11}} - o_k(1)$.

Proof. Define a randomized assignment $F'_U[L']$ as follows: Select a random $V \subseteq U$ conditioned on the event that $L' \subseteq X_V$. Set $F'_U[L'] = \widehat{\mathcal{B}}(V, L')$.

Consider the following two distributions:

Distribution $\mathcal{D}_U$:

- Select $\mathbf{V}$ u.a.r from $\{V \mid (U,V) \in E\}$
- Select $\mathbf{L}'$ u.a.r from $Gr(X_V, \ell - 1)$
- Select $\mathbf{L}$ u.a.r. from $\{L \mid L \in Gr(X_U, \ell) \text{ and } L' \subseteq L\}$
- Let $\mathcal{C}$ be the equivalence class such that $(U, L) \in \mathcal{C}$, select $\mathbf{x} \sim R_{\mathcal{C}}$ as in the edge distribution $\widehat{E}$.
- Select $\mathbf{b} \in \{0,1\}$ u.a.r.

Distribution $\mathcal{D}'_U$:

- Select $\mathbf{L}'$ u.a.r from $Gr(X_U, \ell-1)$
- Select $\mathbf{V}$ u.a.r from $\{V \mid (U,V) \in E \text{ and } L' \in Gr(X_V, \ell-1)\}$
- Select $\mathbf{L}$ u.a.r. from $\{L \mid L \in Gr(X_U,\ell) \text{ and } L' \subseteq L\}$
- Let $\mathcal{C}$ be the equivalence class such that $(U,L) \in \mathcal{C}$, select $\mathbf{x} \sim R_{\mathcal{C}}$ as in the edge distribution $\widehat{E}$.
- Select $\mathbf{b} \in \{0,1\}$ u.a.r.

We have the following lemma from [11].

▶ **Lemma 26** ([11]). *Consider the two marginal distributions on the pair (V,L'), one w.r.t $\mathcal{D}_U$ and another w.r.t $\mathcal{D}'_U$. If $2^\ell \beta \leq \frac{1}{4}$, then the statistical distance between the two distributions is at most $\beta\sqrt{k}\cdot 2^{\ell+3}$.*

In the distribution $\mathcal{D}_U$, there is always constraint between $\mathcal{C}_{x,b}$ and (V,L') in $\mathsf{UG}_{\texttt{folded}}$. Moreover, the distribution of $(\mathcal{C}_{x,b},(V,L'))$ is same as the edge distribution $\widehat{E}$. Therefore

$$p(U) = \Pr_{\mathcal{D}_U}\left[\widehat{\mathcal{A}}(\mathcal{C}_{x,b}), \widehat{\mathcal{B}}(V,L') \text{ satisfy the edge } (\mathcal{C}_{x,b},(V,L'))\right].$$

Rewriting the above equality,

$$p(U) = \Pr_{\mathcal{D}_U}\left[\widehat{\sigma}_U|_{L'} = \widehat{\mathcal{B}}(V,L') \mid \sigma = \widehat{\mathcal{A}}(\mathcal{C}_{x,b})\right].$$

Using Claim 18, the distribution of $F_U[L]$, conditioned on $x \in R_{\mathcal{C}} \setminus (H_U \oplus L')$, is same as the distribution $\widehat{\mathcal{A}}(\mathcal{C}_{x,b})$ (with appropriate unfolding of it) chosen with respect to $\mathcal{D}_U$. As $|R_{\mathcal{C}} \setminus (H_U \oplus L')| = |R_{\mathcal{C}}|/2$ for a random $x \in R_{\mathcal{C}}$, the event $x \in R_{\mathcal{C}} \setminus (H_U \oplus L')$ happens with probability $\frac{1}{2}$. Since we pick an uniformly random $x \in R_{\mathcal{C}}$ while defining $\tilde{\mathcal{A}}(\mathcal{C})$, which in turn defines $F_U[L]$, we have

$$\frac{p(U)}{2} \leqslant \mathop{\mathbf{E}}_{F_U} \Pr_{\mathcal{D}_U}\left[F_U[L]_{|L'} = \widehat{\mathcal{B}}(V,L')\right],$$

Now,

$$\Pr_{\mathcal{D}_U}\left[F_U[L]|_{L'} = \widehat{\mathcal{B}}(V,L')\right] \approx \Pr_{\mathcal{D}'_U}\left[F_U[L]|_{L'} = \widehat{\mathcal{B}}(V,L')\right].$$

follows from the closeness of distributions $\mathcal{D}_U$ and $\mathcal{D}'_U$ on (V,L') given by Lemma 26 by setting $\beta \ll \frac{1}{\sqrt{k}}$ (this setting of β is consistent with the setting of β in Theorem 23). Conditioned on L' the distribution of (V,L') in $\mathcal{D}'_U$ is same as the distribution we used to assign $F'_U[L']$ and therefore we get

$$\frac{p(U)}{2} - o_k(1) \leqslant \mathop{\mathbf{E}}_{F_U} \Pr_{L' \subseteq L}\left[F_U[L]|_{L'} = F'_U[L']\right].$$

Let E_1 be the event that $\frac{p(U)}{4} \leqslant \Pr_{L'\subseteq L}\left[F_U[L]|_{L'} = F'_U[L']\right]$, by averaging argument $\Pr[E_1] \geqslant \frac{P(U)}{4}$. We now fix an F_U for which E_1 occurs. By an averaging argument, there are at least $\frac{p(U)}{8}$ fraction of $L' \in Gr(X_U,\ell-1)$ such that $\Pr_{L\supseteq L'}\left[F_U[L]|_{L'} = F'_U[L']\right] \geqslant \frac{p(U)}{8}$. For each of such L' we have,

$$\Pr_{L_1,L_2\supseteq L'}\left[F_U[L_1] = F_U[L_2]\right] = \Pr_{L_1,L_2\supseteq L'}\left[F_U[L_1]|_{L'} = F_U[L_2]|_{L'} = F'_U[L']\right] \geqslant \frac{p(U)^2}{2^6} - o_k(1).$$

Thus overall, we get

$$\Pr_{L_1, L_2 \supseteq L'}[F_U[L_1] = F_U[L_2] \mid E_1] \geqslant \frac{p(U)^3}{2^9} - o_k(1).$$

Hence,

$$\mathop{\mathbf{E}}_{F_U}[\texttt{agreement}(F_U)] \geqslant \Pr[E_1] \cdot \Pr_{L_1, L_2 \supseteq L'}[F_U[L_1] = F_U[L_2] \mid E_1] \geqslant \frac{p(U)^4}{2^{11}} - o_k(1). \quad \blacktriangleleft$$

There are at least $\frac{\delta}{2}$ fraction of U such that $p(U) \geqslant \frac{\delta}{2}$. This means for at least $\frac{\delta}{2}$ fraction of U, $\mathbf{E}[\texttt{agreement}(F_U)] \geqslant \frac{\delta^4}{2^{15}} - o_k(1)$ using the previous claim. Thus, again by an averaging argument, there exists a fixed $\{F_U : U \in \mathcal{U}\}$, coming from unfolding of some assignment $\tilde{\mathcal{A}}$, such that for at least $\frac{\delta^4}{2^{16}}$ fraction of U, we have $\texttt{agreement}(F_U) \geqslant \frac{\delta^4}{2^{16}}$. The Lemma now follows from Theorem 23. $\blacktriangleleft$

We now prove the main theorem.

Proof of Theorem 2. Fix $\delta > 0$. We let q, β and $\ell \ll k$ be as given in the setting of Theorem 23. Firstly, if we look the the marginal distribution of the edge distribution on $\widehat{A}$ then it is uniform and hence the instance is left-regular.[2] Now, starting with an instance of (X, Eq) we have the following two guarantees of the reduction:

1. If the instance (X, Eq) is $1 - \frac{2\delta}{k}$ satisfiable then by Theorem 20 and Lemma 21, the Unique Games instance $\mathsf{UG}_{\texttt{folded}}$ has a property that for at least $(\frac{1}{2} - \delta)$ fraction of the vertices in $\widehat{A}$, all the edges incident on them are satisfied.
2. Consider the other case in which the instance (X, Eq) is at most $s < 1$, satisfiable. If the Unique Games instance $\mathsf{UG}_{\texttt{folded}}$ is has a δ-satisfying assignment, then by Lemma 24 there is a provers' strategy which can make the outer verifier accepts with probability at least $p_{\frac{\delta^4}{2^{16}}} \gg 2^{-\Omega(\beta k/2^q)}$ for large enough k. This contradicts Theorem 11 and hence in this case, $\mathsf{UG}_{\texttt{folded}}$ has no assignment which satisfies δ fraction of the edges.

Since by Theorem 10 distinguishing between a given instance (X, Eq) being at least $1 - \frac{2\delta}{k}$ satisfiable or at most s satisfiable is NP-hard, this proves our main theorem. $\blacktriangleleft$

5 Independent set in degree d graphs

We consider a weighted graph $H = (V, E)$ where the sum of all weights of all the vertices is 1 and also sum of weights of all the edges is also 1. For $S \subseteq V$, we will denote the total weight of vertices in S by $w(S)$.

▶ **Definition 27.** *A graph H is (δ, ε)-dense if for every $S \subseteq V(H)$ with $w(S) \geqslant \delta$, the total weight of edges inside S is at least ε.*

For $\rho \in [-1, 1]$ and $\beta \in [0, 1]$, the quantity $\Gamma_\rho(\beta)$ is defined as:

$$\Gamma_\rho(\beta) := \Pr[X \leq \phi^{-1}(\beta) \wedge Y \leq \phi^{-1}(\beta)],$$

where X and Y are jointly distributed normal Gaussian random variables with co-variance ρ and ϕ is the cumulative density function of a normal Gaussian random variable.

We will prove the following theorem.

[2] The edges have weights, but it can be made an unweighted left-regular instance by adding multiple edges proportional to its weight with the same constraint.

Let $G(A, B, E, [L], \{\pi_e\}_{e\in E})$ be an instance of Unique Games. The distribution of edges in H is as follows:

- Select $u \in B$ uniformly at random.
- Select its two neighbors v_1 and v_2 uniformly at random. Let π_1 and π_2 are the constraints between (u, v_1) and (u, v_2) respectively.
- Select $x, y \in \{0,1\}^L$, such that for each $i \in [L]$, (x_i, y_i) are sampled independently from the distribution $\mathcal{D}$.
- Output an edge $(v_1, x \circ \pi_1), (v_2, y \circ \pi_2)$.

Figure 3 Reduction from UG to Independent Set from [3].

▶ **Theorem 28.** *Fix $\varepsilon > 0, p \in \left(0, \frac{1}{2}\right]$, then for all sufficiently small $\delta > 0$, there exists a polynomial time reduction from an instance of a left-regular Unique Games $G(A, B, E, [L], \{\pi_e\}_{e\in E})$ to a graph H such that*

1. *If $\mathsf{sval}(G) \geqslant c$, then there is an independent set of weight $c \cdot p$ in H.*
2. *If $\mathsf{val}(G) \leqslant \delta$, then H is $(\beta, \Gamma_\rho(\beta) - \varepsilon)$ dense for every $\beta \in [0,1]$ and $\rho = -\frac{p}{p-1}$.*

The reduction is exactly the same as the one in [3]. We will only show the complete case (1) here. The soundness is proved in [3]. This theorem will imply Theorem 3 using a randomized sparsification technique of [3] to convert the weighted graph into a bounded degree unweighted graph.

5.1 The AKS reduction

Consider the distribution $\mathcal{D}$ on $(a, b) \in \{0,1\}^2$ such that $\Pr[a = b = 1] = 0$ and each bit is p-biased i.e. $\Pr[b = 1] = \Pr[b = 1] = p$. For a string $x \in \{0,1\}^L$ and a permutation $\pi : [L] \to [L]$, let $x \circ \pi \in \{0,1\}^L$, $(x \circ \pi)_i = x_{\pi(i)}$.

Let $G(A, B, E, [L], \{\pi_e\}_{e\in E})$ be an instance of Unique Games which is regular on the A side. We convert it into a weighted graph H. The vertex set is $A \times \{0,1\}^L$. Weight of a vertex (v, x) where $v \in A$ and $x \in \{0,1\}^L$ is $\frac{\mu_p(x)}{|A|}$, where $\mu_p(x) := p^{|x|}(1-p)^{L-|x|}$. The edge distribution is given as follows:

▶ **Lemma 29** (Completeness). *If $\mathsf{sval}(G) \geqslant c$, then there is an independent set in H of weight $c \cdot p$.*

Proof. Fix an assignment $\ell : A \cup B \to \Sigma$ which gives $\mathsf{sval}(G) \geqslant c$. Let $A' \subseteq A$ be the set of vertices such that its edges are satisfied by ℓ, we know that $|A'| \geqslant c \cdot |A|$. Consider the following subset of vertices in H.

$$I = \{(v, x) \mid v \in A', x_{\ell(v)} = 1\}.$$

Firstly, the weight of set I is $c \cdot p$. We show that I is in fact an independent set in H. Suppose for contradiction, there exists an edge $(v_1, x), (v_2, y)$ and both of its endpoints in I. Let u be the common neighbor of v_1, v_2 (one such u must exist). If we let π_1 and π_2 be the permutation constraints between (u, v_1) and (u, v_2) then the conditions for being an edge implies that $(x_{\pi_1(\ell(u))}, y_{\pi_2(\ell(u))})$ should have a support in $\mathcal{D}$. Since all the edges incident on A' are satisfied, $\pi_i(\ell(u)) = \ell(v_i)$ for $i \in \{1, 2\}$. Therefore,$(x_{\ell(v_1)}, y_{\ell(v_2)})$ is also supported in $\mathcal{D}$ and hence both cannot be 1 which implies that both cannot belong to I. ◀

▶ **Lemma 30** (Soundness [3]). *For every $\varepsilon > 0$, if H is not $(\beta, \Gamma_\rho(\beta) - \varepsilon)$-dense for some $\beta \in [0,1]$ and $\rho = -\frac{p}{p-1}$, then G is δ-satisfiable for $\delta := \delta(\varepsilon, p) > 0$.*

Lemma 29 and Lemma 30 prove Theorem 28.

Let $G(A, B, E, [L], \{\pi_e\}_{e\in E})$ be an instance of Unique Games. Fix a graph $H([m], E_H)$ from Lemma 32 with parameters $\eta > 0$ and $t \in \mathbb{Z}^+$, along with the distribution $\mathcal{D}$. Construct a weighted directed graph $\mathcal{G}$ on $B \times [m]^L$ with the following distribution on the edges:

- Select $u \in A$ uniformly at random.
- Select its two neighbors v_1 and v_2 uniformly at random. Let π_1 and π_2 are the constraints between (u, v_1) and (u, v_2) respectively.
- Pick an edge $e = (a, b) \in E_H$ at random from the graph H.
- Select $x, y \in [m]^L$, such that for each $i \in [L]$, (x_i, y_i) are sampled independently as follows:
  - sample $O \sim \mathcal{D}$, set $x_i = O(a)$ and $y_i = O(b)$.
- Perturb x and y as follows: for each $i \in [L]$, with probability $(1 - \varepsilon)$, set $\tilde{x}_i = x_i$, with probability ε set $\tilde{x}_i$ to be u.a.r from $[m]$. Do the same thing for y independently to get $\tilde{y}$.
- Output a directed edge $(v_1, \tilde{x} \circ \pi_1) \to (v_2, \tilde{y} \circ \pi_2)$.

Figure 4 Reduction from UG to Max-Acyclic Graph from [13].

6 Maximum Acyclic Subgraph

In this section we state the reduction from [13] and analyze the completeness case. Given a directed graph $H = (V, E)$, we will denote by $\texttt{Val}(H)$ the fraction of edges in the maximum sized acyclic subgraph of H. We need the following definition.

▶ **Definition 31.** *A t-ordering of a directed graph $H = (V, E)$ consists of a map $O : V \to [t]$. The value of a t-ordering O is given by*

$$\texttt{Val}_t(O) = \Pr_{(a,b)\in E}[O(a) < O(b)] + \frac{1}{2} \cdot \Pr_{(a,b)\in E}[O(a) = O(b)].$$

Define $\texttt{Val}_t(H)$ as:

$$\texttt{Val}_t(H) = \max_O \texttt{Val}_t(O).$$

The following lemma [13] will be crucial in the reduction from Unique Games to Maximum Acyclic Subgraph.

▶ **Lemma 32** ([13]). *Given $\eta > 0$ and a positive integer t, for every sufficiently large m, there exists a weighted directed acyclic graphs $H(V, E)$ on m vertices along with a of distribution $\mathcal{D}$ on the orderings $\{O : V \to [m]\}$ such that:*

1. *For every $u \in V$ and $i \in [m]$, $\Pr_{O\sim\mathcal{D}}[O(u) = i] = \frac{1}{m}$.*
2. *For every directed edge $(a \to b)$, $\Pr_{O\sim\mathcal{D}}[O(a) < O(b)] \geqslant 1 - \eta$.*
3. *$\texttt{Val}_t(H) \leqslant \frac{1}{2} + \eta$.*

The reduction is given in Figure 4. For a string $x \in [q]^L$ and a permutation $\pi : [L] \to [L]$, let $x \circ \pi \in [q]^L$ such that $(x \circ \pi)_i = x_{\pi(i)}$.

▶ **Lemma 33** (Completeness). *For small enough $\varepsilon, \eta > 0$, if the Unique Games instance G has $\mathsf{sval}(G) \geqslant c$ then $\texttt{Val}(\mathcal{G}) \geqslant c \cdot (1 - 2\epsilon)(1 - \eta) + (1 - c) \cdot \left(\frac{1}{2} - \frac{1}{2m}\right)$.*

Proof. Fix an assignment $\ell : A \cup B \to \Sigma$ which gives $\mathsf{sval}(G) \geqslant c$. Let $A' \subseteq A$ be the set of vertices such that its edges are satisfied by ℓ, we know that $|A'| \geqslant c \cdot |A|$. Consider the following m ordering $\mathcal{O} : B \times [m]^L \to [m]$ of the vertices of $\mathcal{G}$: $\mathcal{O}(v, x) = x_{\ell(v)}$. We will show that $\mathtt{Val}_m(\mathcal{O}) \geqslant c(1-\varepsilon)(1-\eta) + (1-c) \cdot \frac{1}{2}$. This will prove the lemma.

$$\begin{aligned}\mathtt{Val}(\mathcal{G}) \geqslant \mathtt{Val}_m(\mathcal{O}) &\geqslant \Pr[\mathcal{O}((v_1, \tilde{x} \circ \pi_1) < \mathcal{O}(v_2, \tilde{y} \circ \pi_2)] \\ &= \Pr[\tilde{x}_{\pi_1(\ell(v_1))} < \tilde{y}_{\pi_2(\ell(v_2))}] \\ &\geqslant c \cdot \Pr[\tilde{x}_{\pi_1(\ell(v_1))} < \tilde{y}_{\pi_2(\ell(v_2))} \mid u \in A'] \\ &\quad + (1-c) \cdot \Pr[\tilde{x}_{\pi_1(\ell(v_1))} < \tilde{y}_{\pi_2(\ell(v_2))} \mid u \notin A']. \end{aligned} \tag{1}$$

Now, if $u \in A'$ then $\pi_1(\ell(v_1)) = \pi_2(\ell(v_2)) = \ell(u)$ and hence,

$$\begin{aligned}\Pr[\tilde{x}_{\pi_1(\ell(v_1))} < \tilde{y}_{\pi_2(\ell(v_2))} \mid u \in A'] &= \Pr[\tilde{x}_{\ell(u)} < \tilde{y}_{\ell(u)}] \\ &\geqslant (1-2\epsilon) \cdot \mathop{\mathbf{E}}_{(a,b) \in E_H} \Pr_{O \sim \mathcal{D}}[O(a) < O(b)] \\ &\geqslant (1-2\epsilon)(1-\eta). \end{aligned} \tag{2}$$

Now, we can lower bound $\Pr[\tilde{x}_{\pi_1(\ell(v_1))} < \tilde{y}_{\pi_2(\ell(v_2))} \mid u \notin A']$ by $(1-2\epsilon)(1-\eta)$ as above if $\pi_1(\ell(v_1)) = \pi_2(\ell(v_2))$. If $\pi_1(\ell(v_1)) \neq \pi_2(\ell(v_2))$ then $\tilde{x}_{\pi_1(\ell(v_1))}$ and $\tilde{y}_{\pi_1(\ell(v_1))}$ are uncorrelated and are distributed uniformly in $[m]$. Therefore, $\Pr[\tilde{x}_{\pi_1(\ell(v_1))} < \tilde{y}_{\pi_2(\ell(v_2))} \mid u \notin A'] = \frac{\binom{m}{2}}{m^2} = \frac{1}{2} - \frac{1}{2m}$. Thus, for small enough ε and η, we can lower bound

$$\Pr[\tilde{x}_{\pi_1(\ell(v_1))} < \tilde{y}_{\pi_2(\ell(v_2))} \mid u \notin A'] \geqslant \min\left\{(1-2\epsilon)(1-\eta), \frac{1}{2} - \frac{1}{2m}\right\} \geqslant \frac{1}{2} - \frac{1}{2m}. \tag{3}$$

Plugging (2) and (3) into (1), we get

$$\mathtt{Val}(\mathcal{G}) \geqslant c \cdot (1-2\epsilon)(1-\eta) + (1-c) \cdot \left(\frac{1}{2} - \frac{1}{2m}\right). \qquad \blacktriangleleft$$

The following soundness of the reduction is shown in [13].

▶ **Lemma 34** (Soundness [13]). *If the Unique Games instance G has $\mathsf{val}(G) \leqslant \delta$ then $\mathtt{Val}(\mathcal{G}) \leqslant \frac{1}{2} + \eta + o_t(1) + \delta'$, where $\delta' \to 0$ as $\delta \to 0$.*

Proof of Theorem 4

For every $\varepsilon' < 0$, setting $\varepsilon, \eta, \delta > 0$ small enough constants and m large enough, in the completeness case we have a maximum acyclic subgraph of size at least $\frac{c}{2} + \frac{1}{2} - \varepsilon'$, whereas in the soundness case it is at most $\frac{1}{2} + \varepsilon'$. Since by Theorem 2, it is NP-hard to distingusih between $\mathsf{sval}(G) \geqslant \frac{1}{2} - \delta$ and $\mathsf{val}(G) \leqslant \delta$ we get that it is NP-hard to approximate the Maximum Acyclic Subgraph problem within a factor of $\frac{1/2+\varepsilon'}{1/4+1/2+\varepsilon'+\delta/2} \approx \frac{2}{3}$.

▶ Remark 35. Instead of $\mathsf{sval}(G) = \frac{1}{2}$, if we only have $\mathsf{val}(G) = \frac{1}{2}$, then the same construction and the labeling from Lemma 33 gives $\mathtt{Val}(\mathcal{G}) \geqslant \frac{5}{8}$. To see this, fix an assignment $\ell : A \cup B \to \Sigma$ which gives $\mathsf{val}(G) \geqslant \frac{1}{2}$. Let α_u denote the fraction of edges attached to u that are satisfied by ℓ. Therefore, we have $\mathsf{val}(G) = \mathbf{E}_{u \in A}[\alpha_u] = \frac{1}{2}$. Using a similar analysis as in the completeness case, we get $\mathtt{Val}(\mathcal{G}) \geqslant \mathbf{E}_{u \in A}[\alpha_u^2 \cdot (1-2\epsilon)] + \mathbf{E}_{u \in A}[(1-\alpha_u^2) \cdot \frac{1}{2}] \geqslant (1-2\epsilon)\,\mathbf{E}[\frac{1}{2} + \frac{\alpha_u^2}{2}]$. By Cauchy-Schwartz inequality $\mathbf{E}[\alpha_u^2] \geqslant (\mathbf{E}[\alpha_u])^2 = \frac{1}{4}$ and hence $\mathtt{Val}(\mathcal{G}) \geqslant (1-2\epsilon) \cdot \frac{5}{8}$. This along with the soundness lemma gives the NP-hardness of $\frac{4}{5}$.

Let $G(A, B, E, [L], \{\pi_e\}_{e\in E})$ be an instance of Unique Games.
- Select $u \in A$ uniformly at random.
- Select k neighbors $\{v_1, v_2, \ldots, v_k\}$ of u uniformly at random. Let π_i be the constraints between (u, v_j) for all $j \in [k]$.
- Select $x^1, x^2, \ldots, x^k \in [q]^L$, such that for each $i \in [L]$ sample $(x^1_i, x^2_i, \ldots, x^k_i)$ independently as follows:
  - with probability $(1-\varepsilon)$, $(x^1_i, x^2_i, \ldots, x^k_i)$ is sampled from the distribution $\mathcal{D}$.
  - with probability ε, $(x^1_i, x^2_i, \ldots, x^k_i)$ is sampled from $[q]^k$ uniformly at random.
- Output $((v_1, x^1 \circ \pi_1), (v_2, x^2 \circ \pi_2), \ldots, (v_k, x^k \circ \pi_k))$.

Figure 5 Reduction from UG to a P-CSP instance $\mathcal{I}$ from [5].

7 Predicates supporting Pairwise Independence

In this section, we prove Theorem 5.

7.1 The Austrin-Mossel reduction

Let $\mathcal{D}$ be a distribution on $P^{-1}(1)$ which is balanced and pairwise independent. For a string $x \in [q]^L$ and a permutation $\pi : [L] \to [L]$, let $x \circ \pi \in [q]^L$ such that $(x \circ \pi)_i = x_{\pi(i)}$.

Let $G(A, B, E, [L], \{\pi_e\}_{e\in E})$ be an instance of Unique Games. We convert it into a P-CSP instance $\mathcal{I}$ as follows. The variable set is $B \times [q]^L$. The variable sets are *folded* in the sense that for every assignment $f : B \times [q]^L \to [q]$ to the variables, we enforce that for every $v \in B$, $x \in [q]^L$ and $\alpha \in [q]$,

$$f(v, x + \alpha^L) = f(v, x) + \alpha,$$

where additions are (mod q).

The distribution on the constraints is given in Figure 5:

▶ **Lemma 36** (Completeness). *If* $\mathsf{sval}(G) \geqslant c$, *the* $\mathcal{I}$ *is* $(c-\varepsilon)$*- satisfiable.*

Proof. Fix an assignment $\ell : A \cup B \to \Sigma$ which gives $\mathsf{sval}(G) \geqslant c$. Let $A' \subseteq A$ be the set of vertices such that its edges are satisfied by ℓ, we know that $|A'| \geqslant c \cdot |A|$. Thus with probability c, $u \in A'$ and all edges attached to it are satisfied by ℓ. Consider the following assignment f to the variables of $\mathcal{I}$: For a variable (v, x), we assign $f(v, x) = x_{\ell(v)}$.

Conditioned on $u \in A'$, we will show that $(f(v_1, x^1 \circ \pi_1), f(v_2, x^2 \circ \pi_2), \ldots, f(v_k, x^k \circ \pi_k)) \in P^{-1}(1)$ with probability $(1-\varepsilon)$ and this will prove the lemma. Now, $(f(v_2, x^2 \circ \pi_2), \ldots, f(v_k, x^k \circ \pi_k)))$ is same as $((x^1 \circ \pi_1)_{\ell(v_1)}, (x^2 \circ \pi_2)_{\ell(v_2)}, \ldots, (x^k \circ \pi_k)_{\ell(v_k)})$, which in turns equals $(x^1_{\pi_1(\ell(v_1))}, x^2_{\pi_2(\ell(v_2))}, \ldots, x^k_{\pi_k(\ell(v_k))})$. Since ℓ satisfies all the edges (u, v_i), we have that for all $j \in [k]$, $\pi_j(\ell(v_j)) = \ell(u) =: i$ for some $i \in [L]$. Therefore we get $(x^1_i, x^2_i, \ldots, x^k_i)$, and according to the distribution, it belongs to $P^{-1}(1)$ with probability $(1-\varepsilon)$. ◀

We have the following soundness of the reduction.

▶ **Lemma 37** (Soundness [5]). *If the instance* $\mathcal{I}$ *is* $\frac{P^{-1}(1)}{q^k} + \eta$ *satisfiable, then* G *is* $\delta := \delta(\eta, \varepsilon, k, q) > 0$ *satisfiable.*

The completeness and soundness of the reduction, along with our main theorem, imply Theorem 5.

References

1 Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof Verification and the Hardness of Approximation Problems. *J. ACM*, 45(3):501–555, May 1998. (Preliminary version in *33rd FOCS*, 1992).

2 Sanjeev Arora and Shmuel Safra. Probabilistic Checking of Proofs: A New Characterization of NP. *jacm*, 45(1):70–122, January 1998. (Preliminary version in *33rd FOCS*, 1992).

3 Per Austrin, Subhash Khot, and Muli Safra. Inapproximability of Vertex Cover and Independent Set in Bounded Degree Graphs. *Theory of Computing*, 7(3):27–43, 2011.

4 Per Austrin, Rajsekar Manokaran, and Cenny Wenner. On the NP-Hardness of Approximating Ordering-Constraint Satisfaction Problems. *Theory of Computing*, 11(10):257–283, 2015.

5 Per Austrin and Elchanan Mossel. Approximation Resistant Predicates from Pairwise Independence. *computational complexity*, 18(2):249–271, June 2009.

6 N. Bansal, A. Gupta, and G. Guruganesh. On the Lovász Theta Function for Independent Sets in Sparse Graphs. *SIAM Journal on Computing*, 47(3):1039–1055, 2018.

7 Nikhil Bansal and Subhash Khot. Inapproximability of Hypergraph Vertex Cover and Applications to Scheduling Problems. In *Automata, Languages and Programming, 37th International Colloquium, ICALP 2010, Bordeaux, France, July 6-10, 2010, Proceedings, Part I*, pages 250–261, 2010.

8 Amey Bhangale, Rajiv Gandhi, Mohammad Taghi Hajiaghayi, Rohit Khandekar, and Guy Kortsarz. Bi-Covering: Covering Edges with Two Small Subsets of Vertices. *SIAM J. Discrete Math.*, 31(4):2626–2646, 2017.

9 Siu On Chan. Approximation resistance from pairwise-independent subgroups. *Journal of the ACM (JACM)*, 63(3):27, 2016.

10 Irit Dinur, Subhash Khot, Guy Kindler, Dor Minzer, and Muli Safra. On non-optimally expanding sets in Grassmann graphs. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 940–951. ACM, 2018.

11 Irit Dinur, Subhash Khot, Guy Kindler, Dor Minzer, and Muli Safra. Towards a proof of the 2-to-1 games conjecture? In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018*, pages 376–389, 2018.

12 Uriel Feige, Shafi Goldwasser, Laszlo Lovász, Shmuel Safra, and Mario Szegedy. Interactive proofs and the hardness of approximating cliques. *Journal of the ACM (JACM)*, 43(2):268–292, 1996.

13 Venkatesan Guruswami, Rajsekar Manokaran, and Prasad Raghavendra. Beating the random ordering is hard: Inapproximability of maximum acyclic subgraph. In *Foundations of Computer Science, 2008. FOCS'08. IEEE 49th Annual IEEE Symposium on*, pages 573–582. IEEE, 2008.

14 Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing, STOC 2002*, pages 767–775. ACM, 2002.

15 Subhash Khot, Guy Kindler, Elchanan Mossel, and Ryan O'Donnell. Optimal inapproximability results for MAX-CUT and other 2-variable CSPs? *SIAM Journal on Computing*, 37(1):319–357, 2007.

16 Subhash Khot, Dor Minzer, and Muli Safra. On independent sets, 2-to-2 games, and Grassmann graphs. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017*, pages 576–589, 2017.

17 Subhash Khot, Dor Minzer, and Muli Safra. Pseudorandom Sets in Grassmann Graph have Near-Perfect Expansion. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018*, pages 592–601, 2018.

18 Subhash Khot and Assaf Naor. Approximate kernel clustering. *Mathematika*, 55(1-2):129–165, 2009.

19 Subhash Khot and Oded Regev. Vertex cover might be hard to approximate to within 2- ε. *Journal of Computer and System Sciences*, 74(3):335–349, 2008.

20 Subhash Khot, Madhur Tulsiani, and Pratik Worah. A characterization of strong approximation resistance. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 634–643. ACM, 2014.

21 Subhash A Khot and Nisheeth K Vishnoi. The Unique Games Conjecture, Integrality Gap for Cut Problems and Embeddability of Negative-Type Metrics into ℓ_1. *Journal of the ACM (JACM)*, 62(1):8, 2015.

22 Alantha Newman. The maximum acyclic subgraph problem and degree-3 graphs. In *Approximation, Randomization, and Combinatorial Optimization: Algorithms and Techniques*, pages 147–158. Springer, 2001.

23 Prasad Raghavendra. Optimal algorithms and inapproximability results for every CSP? In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 245–254. ACM, 2008.

24 Ran Raz. A parallel repetition theorem. *SIAM Journal on Computing*, 27(3):763–803, 1998.

Simple and Efficient Pseudorandom Generators from Gaussian Processes

Eshan Chattopadhyay
Cornell University, Ithaca, NY, USA
eshanc@cornell.edu

Anindya De
University of Pennsylvania, Philadelphia, PA, USA
anindyad@seas.upenn.edu

Rocco A. Servedio
Columbia University, New York, NY, USA
rocco@cs.columbia.edu

Abstract

We show that a very simple pseudorandom generator fools intersections of k linear threshold functions (LTFs) and arbitrary functions of k LTFs over n-dimensional Gaussian space. The two analyses of our PRG (for intersections versus arbitrary functions of LTFs) are quite different from each other and from previous analyses of PRGs for functions of halfspaces. Our analysis for arbitrary functions of LTFs establishes bounds on the Wasserstein distance between Gaussian random vectors with similar covariance matrices, and combines these bounds with a conversion from Wasserstein distance to "union-of-orthants" distance from [5]. Our analysis for intersections of LTFs uses extensions of the classical Sudakov-Fernique type inequalities, which give bounds on the difference between the expectations of the maxima of two Gaussian random vectors with similar covariance matrices.

For all values of k, our generator has seed length $O(\log n) + \mathrm{poly}(k)$ for arbitrary functions of k LTFs and $O(\log n) + \mathrm{poly}(\log k)$ for intersections of k LTFs. The best previous result, due to [14], only gave such PRGs for arbitrary functions of k LTFs when $k = O(\log \log n)$ and for intersections of k LTFs when $k = O(\frac{\log n}{\log \log n})$. Thus our PRG achieves an $O(\log n)$ seed length for values of k that are exponentially larger than previous work could achieve.

By combining our PRG over Gaussian space with an invariance principle for arbitrary functions of LTFs and with a regularity lemma, we obtain a deterministic algorithm that approximately counts satisfying assignments of arbitrary functions of k general LTFs over $\{0,1\}^n$ in time $\mathrm{poly}(n) \cdot 2^{\mathrm{poly}(k, 1/\varepsilon)}$ for all values of k. This algorithm has a $\mathrm{poly}(n)$ runtime for $k = (\log n)^c$ for some absolute constant $c > 0$, while the previous best $\mathrm{poly}(n)$-time algorithms could only handle $k = O(\log \log n)$. For intersections of LTFs, by combining these tools with a recent PRG due to [28], we obtain a deterministic algorithm that can approximately count satisfying assignments of intersections of k general LTFs over $\{0,1\}^n$ in time $\mathrm{poly}(n) \cdot 2^{\mathrm{poly}(\log k, 1/\varepsilon)}$. This algorithm has a $\mathrm{poly}(n)$ runtime for $k = 2^{(\log n)^c}$ for some absolute constant $c > 0$, while the previous best $\mathrm{poly}(n)$-time algorithms for intersections of k LTFs, due to [14], could only handle $k = O(\frac{\log n}{\log \log n})$.

2012 ACM Subject Classification Theory of computation → Pseudorandomness and derandomization

Keywords and phrases Polynomial threshold functions, Gaussian processes, Johnson-Lindenstrauss, pseudorandom generators

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.4

Related Version A full version of this paper is available at `https://eccc.weizmann.ac.il/report/2018/100/`.

Funding *Eshan Chattopadhyay*: Supported by NSF grants CCF-1412958, CCF-1849899, and the Simons foundation. Part of the work done while the author was a postdoctoral researcher at the Institute for Advanced Study, Princeton.

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka; Article No. 4; pp. 4:1–4:33

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

Anindya De: Supported by NSF CCF 1926872 (transferred from CCF 1814706). Work done while the author was at Northwestern University supported by a start-up grant.
Rocco A. Servedio: Supported by NSF CCF 1814873, NSF CCF 1563155, and by the Simons Collaboration on Algorithms and Geometry.

Acknowledgements The authors thank Oded Regev and Li-Yang Tan for helpful discussions.

1 Introduction

Constructing explicit pseudorandom generators (PRGs) for interesting classes of Boolean-valued functions is a fundamental problem in complexity theory which has witnessed a rich line of work. An important class of functions, which have been intensively studied from this perspective, are *linear threshold functions* (henceforth referred to as LTFs), i.e. functions of the form $f(x) = \text{sign}(\sum_{i=1}^{n} w_i x_i - \theta)$ for some $w \in \mathbb{R}^n$ and $\theta \in \mathbb{R}$. LTFs arise naturally in a variety of areas including machine learning, social choice theory, circuit complexity and pseudorandomness. Through a very successful line of work [8, 26, 13], explicit PRGs have been obtained which ε-fool the class of LTFs over $\{-1,1\}^n$ with seed length $O(\log n + \log^2(1/\varepsilon))$ [26], or alternately seed length $O(\log(n/\varepsilon)(\log\log(n/\varepsilon))^2)$ [13]. For LTFs over the Gaussian distribution, [23] give an ε-PRG that fools LTFs with seed length $O(\log n + \log(1/\varepsilon)\log\log(1/\varepsilon)))$.

Given these successes in designing PRGs to fool a single LTF, a natural next goal is to develop PRGs for *intersections* of k LTFs (i.e. polytopes with k facets) or, more generally, for arbitrary Boolean functions of k LTFs. PRGs for polytopes have direct applications to central problems at the intersection of derandomization and combinatorial optimization, such as deterministic approximate volume estimation for polytopes and approximate counting of feasible solutions to 0-1 integer programs. The standard way to use a PRG for such applications is to run through the list of all seeds, and hence it is desirable to have seed length as small as possible as a joint function of n and k. In particular, a seed length of the form $O(\log n) \cdot \alpha(k, 1/\varepsilon)$ leads to a running time of $n^{O(\alpha(k,1/\varepsilon))}$, which even for constant ε is super-polynomial for any super-constant k. In contrast, a seed length of the form $O(\log n) + \alpha(k, 1/\varepsilon)$ leads to a running time of $\text{poly}(n) \cdot 2^{\alpha(k,1/\varepsilon)}$, which can be a fixed polynomial in n even for various super-constant values of k (depending on the function α).

In this paper we work over Gaussian space, and we give the first PRGs for intersections and arbitrary functions of k LTFs which have seed length of the form $O(\log n) + \alpha(k, 1/\varepsilon)$ for all k. For intersections of LTFs we achieve $\alpha(k, 1/\varepsilon) = \text{poly}(\log k, 1/\varepsilon)$, and for arbitrary functions of LTFs we achieve $\alpha(k, 1/\varepsilon) = \text{poly}(k, 1/\varepsilon)$. Thus for constant ε our seed length is $O(\log n)$ for $k = 2^{(\log n)^c}$ LTFs (for intersections) and $k = (\log n)^c$ LTFs (for arbitrary functions), where $c > 0$ is an absolute constant. Previously, such an $O(\log n)$ seed length was only known for $k = O(\log(n)/\log\log n)$ (for intersections) and $k = O(\log\log n)$ (for arbitrary functions) [14]. Thus, in both cases our PRGs achieve the (optimal) $O(\log n)$ seed length for exponentially larger values of k than was previously known.

Before stating our results in detail we recall the definition of a PRG over Gaussian space (see [18, 17, 16, 19, 23, 21]):

▶ **Definition 1** (PRGs for Boolean-valued functions over Gaussian space)**.** *Let $\mathcal{C}$ be a class of functions from $\mathbb{R}^n$ to $\{-1,1\}$. Given $\varepsilon > 0$, a function $\mathcal{G} : \{-1,1\}^s \to \mathbb{R}^n$ is an* ε-PRG for class $\mathcal{C}$ over Gaussian space *if for every function $F \in \mathcal{C}$,*

$$|\,\boldsymbol{Pr}[F(\mathcal{G}(\mathbf{U}^{(s)})) = 1] - \boldsymbol{Pr}[F(\mathbf{G}^{(n)}) = 1]| \leq \varepsilon,$$

where $\mathbf{G}^{(n)}$ *denotes* $(\mathbf{G}_1, \ldots, \mathbf{G}_n)$*, a random variable distributed according to the standard Gaussian* $\mathcal{N}(0,1)^n$*, and* $\mathbf{U}^{(s)}$ *denotes the uniform distribution on* $\{-1,1\}^s$*. The parameter* s *is called the* seed length *of* $\mathcal{G}$*.*

1.1 Our results and comparison to prior work

Our PRG results. The following are our main PRG theorems:

▶ **Theorem 2** (Fooling arbitrary functions of LTFs)**.** *There is an explicit PRG which* ε*-fools any Boolean function of* k *LTFs* $g(h_1, \ldots, h_k)$ *over* $\mathcal{N}(0,1)^n$*, for any* $\varepsilon > 0$ *and any* k*, with seed length*

$$O\left(\log n + \mathrm{poly}(k, 1/\varepsilon)\right).$$

This seed length is not far from the best possible in terms of its dependence on both n and k, as it is not difficult (see Appendix A) to establish a seed length lower bound for this class of $\max\{\lfloor \log n \rfloor, k\} = \Omega(k + \log n)$ for any $1 \le k \le n$.

In the special case when the combining function g is an AND, we get an exponential improvement in the seed length dependence on k:[1]

▶ **Theorem 3** (Fooling intersections of LTFs)**.** *There is an explicit PRG which* ε*-fools any intersection of* k *LTFs over* $\mathcal{N}(0,1)^n$*, for any* $\varepsilon > 0$ *and any* k*, with seed length*

$$O\left(\log n + \mathrm{poly}(\log k, 1/\varepsilon)\right).$$

Here too the seed length is not far from best possible for a broad range of parameters; we note that the above-mentioned lower bound of $\log n$ even when $k = 1$ implies a seed length lower bound of $\Omega(\log n)$, which is $\Omega(\log n + \log k)$ for any $k \le \mathrm{poly}(n)$ (the most interesting regime for Theorem 3).

For arbitrary functions of k LTFs, Theorem 2 is the first result which gives a seed length of $O(\log n)$ for $k = (\log n)^c$, and for intersections of k LTFs Theorem 3 is the first result which gives a seed length of $O(\log n)$ for $k = 2^{(\log n)^c}$. As mentioned earlier and discussed in more detail below, an optimal seed length of $O(\log n)$ was previously only known [14] for exponentially smaller values of k in both settings. Below we briefly review prior results on explicit PRGs for these classes, starting with intersections of LTFs.

The most directly comparable prior result for intersections of k LTFs is the main result of [28], which gives a PRG for intersections of k LTFs over $\{-1,1\}^n$ with seed length $\log(n) \cdot \mathrm{poly}(\log k, 1/\varepsilon)$. (Such a PRG directly implies a PRG for intersections of k LTFs over Gaussian space with the same seed length via a standard reduction.) The [28] PRG builds on a PRG due to Harsha et al. [16] which has seed length $\log(n) \cdot \mathrm{poly}(\log k, 1/\varepsilon)$ for intersections of sufficiently regular LTFs; the [16] PRG in turn is similar to a PRG construction of Meka and Zuckerman [26] (for a single LTF) in which the basic idea is to (pseudorandomly) hash the coordinates into buckets and use ℓ-wise independence for coordinates hashed to the same bucket. The analysis of the [28] PRG combines a range of technical ingredients including an invariance principle for polytopes that [16] establish, combinatorial PRGs for depth-2 circuits, and new Littlewood-Offord type theorems for polytopes.

[1] We note that a weak form of Theorem 2, with a seed length of $O\left(\log n + \mathrm{poly}(2^k, 1/\varepsilon)\right)$, follows immediately from Theorem 3 just by setting its error parameter to be $\varepsilon/2^k$ and observing that any function of k LTFs is a union of at most 2^k many disjoint intersections of k LTFs. However, this is exponentially worse than we achieve in Theorem 2 above.

PRGs for intersection of LTFs were also studied by Gopalan, O'Donnell, Wu, and Zuckerman [14], Diakonikolas, Kane and Nelson [9], and recently by Servedio and Tan [29]. These results give PRGs with respect to the uniform distribution on the Boolean cube (in fact, the PRG in [14] fools arbitrary product distributions). For general k, the seed length of the PRG in [14] for intersection of k LTFs is $O((\log n + k\log(k/\varepsilon)) \cdot \log(k/\varepsilon))$. This linear dependence of the seed length on k is far from optimal; for example, if $k \geq n$ then their result does not yield a non-trivial PRG. For the special case when k/ε is at most $\mathrm{poly}(\log n)$, [14] achieves the better seed length of $O(\log n + k\log(k/\varepsilon))$. Thus, for $k = O(\log n/\log\log n)$, the [14] seed length is $O(\log n)$.

The work of Diakonikolas et al. [9] achieves a similar polynomial dependence on k in the seed length of their PRG (more precisely, they achieve seed length $O(\log n \cdot \mathrm{poly}(k, 1/\varepsilon))$, and their PRG works also for intersections of k degree-2 polynomial threshold functions). The work of Servedio and Tan [29] achieves seed length with polylogarithmic dependence on k, but only gives a good bound against intersections of LTFs with small integer weights. In more detail, if each of the k LTFs in the intersection has all its weights w_i being integers in $[-t, t]$, then the PRG in [29] has seed length $\mathrm{poly}(\log n, \log k, t, 1/\varepsilon)$. The parameter t for an LTF can in general be exponential in n (and in fact, for a random LTF, t is exponential in n with high probability), and hence the [29] result is of interest only for intersections of low-weight LTFs.

Turning to arbitrary functions of k LTFs, we observe that (as indicated in the earlier footnote) any PRG for intersections of k LTFs can be used to fool arbitrary functions of k LTFs by setting its accuracy parameter to $\varepsilon/2^k$. If the seed length of the PRG has an inverse polynomial dependence on the accuracy parameter (as in our result) then this simple approach does not yield a very good seed length, but [14] used essentially this approach to obtain a PRG that fools any function of k LTFs with seed length $O((k^2 + k\log(1/\varepsilon) + \log n) \cdot (k + \log(1/\varepsilon)))$. In the special case when $k \cdot 2^k/\varepsilon$ is at most $\mathrm{poly}(\log n)$, they achieve a better seed length of $O(k^2 + k\log(1/\varepsilon) + \log n)$, which is $O(\log n)$ for constant ε and $k = O(\log\log n)$.

Our results on deterministic approximate counting. By combining our new PRGs with invariance principles and a (multi-)regularity lemma, we obtain deterministic algorithms which approximately count the number of satisfying assignments to intersections or arbitrary functions of k arbitrary LTFs over $\{-1,1\}^n$. (Note that such algorithms, unlike PRGs, are non-oblivious, i.e. they can "inspect" the particular LTFs which comprise the input to the problem.)

▶ **Theorem 4** (Deterministic approximate counting for arbitrary functions of k LTFs over $\{-1,1\}^n$)**.** *There is a deterministic algorithm which, given as input k LTFs $h_1, \ldots, h_k$ over $\{-1,1\}^n$, an explicit function $g : \{-1,1\}^k \to \{-1,1\}$ and an error parameter $\varepsilon > 0$, runs in $\mathrm{poly}(n) \cdot 2^{\mathrm{poly}(k,1/\varepsilon)}$ time and outputs a value $\tilde{v} \in [0,1]$ such that $|\tilde{v} - v| \leq \varepsilon$, where v is the fraction of points in $\{-1,1\}^n$ that satisfy $g(h_1, \ldots, h_k)$.*

For intersections of LTFs, by combining our approach with the [28] PRG we can get an exponentially better runtime dependence on k:

▶ **Theorem 5** (Deterministic approximate counting for intersections of k LTFs over $\{-1,1\}^n$)**.** *There is a deterministic algorithm which, given as input k LTFs $h_1, \ldots, h_k$ over $\{-1,1\}^n$ and an error parameter $\varepsilon > 0$, runs in $\mathrm{poly}(n) \cdot 2^{\mathrm{poly}(\log k, 1/\varepsilon)}$ time and outputs a value $\tilde{v} \in [0,1]$ such that $|\tilde{v} - v| \leq \varepsilon$, where v is the fraction of points in $\{-1,1\}^n$ that satisfy $h_1 \wedge \cdots \wedge h_k$.*

We are not aware of prior results on deterministic approximate counting for intersections (or arbitrary functions) of k LTFs which run faster than simply enumerating over the seeds of a PRG. Thus Theorem 4 gives the first deterministic algorithm that runs in *fixed*

poly(n) runtime even for k which is polylogarithmic in n; as indicated earlier, given the previous state of the art on PRGs for arbitrary functions of k LTFs for $k = \omega(\log\log n)$ prior algorithms would have a running time of at least $n^{\mathrm{poly}(k)}$. Similarly, Theorem 5 gives the first deterministic algorithm that runs in *fixed* poly(n) runtime even for $k = 2^{(\log n)^{\Omega(1)}}$. The previous state of the art on PRGs for intersection of k LTFs for $k = \omega(\log n/\log\log n)$ would have a running time of at least $n^{\mathrm{poly}(\log k)}$ (such a running time is obtained simply by enumerating over the seeds of the [28] PRG).

A key ingredient in the proof of Theorem 4 is an invariance principle for *arbitrary functions* of k LTFs, analogous to the main structural result of [16] which is an invariance principle for *intersections* of k LTFs. Such an invariance principle was proved in [14], and we provide an alternate proof in Appendix C (which is very different from the proofs of the invariance principles in [16, 14]). We believe this could be of independent interest. We elaborate on this, still at a conceptual level, in Section 3 and give full details in Section 7.

A straightforward approach to Theorem 5 using only the multi-regularity lemma and an invariance principle would have a running time which is exponential in k because the number of leaves in the decision tree constructed by the multi-regularity lemma is exponential in k. We achieve a quasi-polynomial dependence on k by exploiting additional structure in the decision tree (specifically, that it is a so-called "junta decision tree" in which the same variable occurs at each node of any given depth). Intuitively, this makes it possible for us to use the [28] PRG on the space of all variables occurring in the decision tree (to "pseudorandomly sample" leaves of the decision tree and use only those to construct an accurate estimate of the overall desired probability). Since the size of this variable space, roughly speaking, is $m = \tilde{O}(k)$ (crucially with no dependence on n), the [28] PRG's seed length in this context (of intersections of k LTFs over m variables) is $\log(m) \cdot \mathrm{poly}(\log k, 1/\varepsilon) = \mathrm{poly}(\log k, 1/\varepsilon)$, which leads to our overall final running time of $\mathrm{poly}(n) \cdot 2^{\mathrm{poly}(\log k, 1/\varepsilon)}$.

2 Our PRG and a high-level overview of its analysis

We use the same simple PRG construction to obtain both of our PRG results (Theorems 2 and 3); the two results are obtained by instantiating the parameters in two different ways. We describe this PRG below with general parameters; the precise parameter settings we use for each class (intersections versus arbitrary functions of k LTFs) will be made clear in the course of the respective analyses.

An idealized version of our PRG is as follows:

1. Let $\mathbf{G}^{(d)}$ be an $\mathcal{N}(0,1)^d$ Gaussian (which we view as a column vector).
2. Let $\mathbf{A} \in \mathbb{R}^{d \times n}$ be a pseudorandom Johnson-Lindenstrauss matrix drawn from the distribution of pseudorandom $d \times n$ JL-matrices given by the work of Kane, Meka and Nelson [20] (more details on this will be given below).
3. A draw from our generator Gen is $\mathbf{Z} := \mathbf{A}^{\mathsf{T}}\mathbf{G}^{(d)}$ (note that this is a vector in $\mathbb{R}^n$).

The actual PRG differs from the above-described idealized version because using finitely many bits it is not possible to generate a draw from the continuous $\mathbf{G}^{(d)}$ distribution with perfect fidelity. So in Step 1 the actual PRG uses a discrete approximation of each coordinate of $\mathbf{G}^{(d)}$ (we explain precisely what is meant by this in Appendix B); let $\hat{\mathbf{G}}^{(d)}$ denote the resulting distribution over $\mathbb{R}^d$. For clarity of exposition, the main analysis in the paper will be carried out for a "perfect" Gaussian $\mathbf{G}^{(d)}$, i.e. we will analyze the idealized PRG and show that it is an $O(\varepsilon)$-PRG for each of our two classes of interest (intersections and arbitrary functions of k LTFs). Appendix B shows that, for each of these two classes, if the idealized generator (which uses $\mathbf{G}^{(d)}$) is an $O(\varepsilon)$-PRG, then so is the actual generator which uses $\hat{\mathbf{G}}^{(d)}$.

High level idea of our generator. The Johnson-Lindenstrauss (JL) transform is one of the most important tools in high-dimensional data analysis. In a nutshell, for any error parameter ε, the JL transform gives a family $\mathcal{D}$ of $d \times n$ matrices such that for $\mathbf{A} \sim \mathcal{D}$ and any k unit vectors $W^1, \ldots, W^k \in \mathbb{R}^n$, with high probability, the following holds: For all $0 \leq i, j \leq k$, $\|\mathbf{A}W^i - \mathbf{A}W^j\|_2 = (1 \pm \varepsilon)\|W^i - W^j\|_2$ (where $W^0 = 0$). Crucially, one can obtain this guarantee with $d = O(\varepsilon^{-2} \log k)$.

We can reinterpret the guarantee of the JL transform in the following way: Let $\mathbf{A} \sim \mathcal{D}$ and consider the two distributions $\mathbf{Z} := \mathbf{A}^T \cdot \mathbf{G}^{(d)}$ and $\mathbf{Z}' = \mathbf{G}^{(n)}$. Let $W \in \mathbb{R}^{k \times n}$ be the $k \times n$ matrix whose rows are $W^1, \ldots, W^k$. Then, for any $\vec{\theta}$, the distributions $\mathbf{X} = W \cdot \mathbf{Z} - \vec{\theta}$ and $\mathbf{Y} = W \cdot \mathbf{Z}' - \vec{\theta}$ (i) are both Gaussian distributions over $\mathbb{R}^k$, (ii) have the same mean, and (iii) are such that the two $k \times k$ covariance matrices $\mathbf{Cov}(\mathbf{X})$ and $\mathbf{Cov}(\mathbf{Y})$ differ pointwise by at most ε. Let us define the affine function $f : \mathbb{R}^n \to \mathbb{R}^k$ as $f(z) = Wz - \vec{\theta}$. Then, the guarantee of the JL transform is that $\mathsf{Cov}(f(\mathbf{Z})) \approx_\varepsilon \mathsf{Cov}(f(\mathbf{Z}'))$; we may loosely view this guarantee as showing that the generator above *fools the covariance*.

The above perspective leads to the insight which motivates our work, which is essentially the following: since both $\mathbf{X}$ and $\mathbf{Y}$ are Gaussians, which are completely determined by their means and covariances, *other interesting tests besides the covariance may reasonably be expected to be fooled by (a pseudorandom version of) the Johnson-Lindenstrauss transform.* In this paper we consider tests of the form $h(\mathrm{sign}(f(z)_1), \ldots, \mathrm{sign}(f(z)_k))$, where h may be any function from $\{-1, 1\}^k$ to $\{-1, 1\}$ (we will also specialize to the case where h is an AND) and $f(z)_i$ denotes the i^{th} coordinate of $f(z)$. In other words, we are interested in fooling functions (given by h) of k LTFs (given by $\mathrm{sign}(f(z)_1), \ldots, \mathrm{sign}(f(z)_k)$). As we show in this paper, for a suitable choice of d (depending on whether h is arbitrary or is an AND) our generator can indeed fool all functions of the above form.

Seed length of our PRG. In order to state the seed length of our generator we first need to identify all of the relevant parameters. In Step 1, for each of our two results we will take $d = O(\log(k/\delta')/\varepsilon'^2)$ where ε' is a parameter that will be discussed below; as mentioned above each coordinate of $\hat{\mathbf{G}}^{(d)}$ will be a discrete approximation of an $\mathcal{N}(0, 1)$ Gaussian. In Step 2, the KMN distribution over pseudorandom $d \times n$ JL-matrices has two additional parameters, which we denote ε' and δ' (see Section 4.2 for details.)

For the first step, as we show in Appendix B, a total of $O(d \log(kd/\varepsilon))$ many random bits suffice to generate a draw from $\hat{\mathbf{G}}^{(d)}$. For the second step, as we discuss in Section 4.2, a pseudorandom $d \times n$ JL-matrix with parameters ε', δ' can be drawn from the KMN distribution using $O(\log n + \log(1/\delta') \cdot \log(\log(1/\delta')/\varepsilon'))$ bits of randomness. So the overall seed length of our PRG is

$$\begin{aligned} &O(d \log(kd/\varepsilon)) + O(\log n + \log(1/\delta') \cdot \log(\log(1/\delta')/\varepsilon')) \\ &= O\left(\frac{\log(k/\delta')}{\varepsilon'^2} \cdot (\log k + \log\log(k/\delta') + \log(1/(\varepsilon'\varepsilon))) + \log n\right). \end{aligned}$$

As we will see in Section 4.2, we will always take δ' to be ε, so the seed length of our generator is

$$O\left(\frac{\log(k/\varepsilon)}{\varepsilon'^2} \cdot (\log k + \log\log(k/\varepsilon) + \log(1/(\varepsilon'\varepsilon))) + \log n\right). \tag{1}$$

We will instantiate the parameter ε' to one specific value (a function of k and ε) in Section 5 for arbitrary functions of LTFs, and to another specific value in Section 6 for intersections of LTFs, thus obtaining the seed lengths claimed in Theorems 2 and 3.

In the rest of this section we give an overview of the analyses of our PRGs. While the same PRG gives both our results, the analyses are quite different for the two classes we consider (arbitrary functions of LTFs and intersections of LTFs). We first sketch the (simpler) analysis for fooling arbitrary functions of LTFs.

2.1 An outline of our analysis for fooling arbitrary functions of LTFs

We start by recalling some definitions which are useful for our overview. An *orthant* of $\mathbb{R}^k$ is a subset $O \subset \mathbb{R}^k$ of the form

$$O = \{x \in \mathbb{R}^k : \mathrm{sign}(x_i) = b_i, i = 1, \ldots, k\} \qquad \text{for some } (b_1, \ldots, b_k) \in \{-1, 1\}^k.$$

Given two random variables $\mathbf{X}, \mathbf{Y}$ over $\mathbb{R}^k$, the *quadratic Wasserstein distance* $\mathcal{W}_2(\mathbf{X}, \mathbf{Y})$ between $\mathbf{X}$ and $\mathbf{Y}$ is defined to be

$$\mathcal{W}_2(\mathbf{X}, \mathbf{Y}) = \inf_{(\widehat{\mathbf{X}}, \widehat{\mathbf{Y}})} (\mathbf{E}[\|\widehat{\mathbf{X}} - \widehat{\mathbf{Y}}\|^2])^{1/2},$$

where the infimum is taken over all couplings $(\widehat{\mathbf{X}}, \widehat{\mathbf{Y}})$ of $\mathbf{X}$ and $\mathbf{Y}$.[2]

Now we can present our overview. Our goal is to show that our PRG ε-fools every function of the form $g(h_1(x), \ldots, h_k(x)) : \mathbb{R}^n \to \{-1, 1\}$, where $g : \{-1, 1\}^k \to \{-1, 1\}$ is arbitrary and each $h_i : \mathbb{R}^n \to \{-1, 1\}$ is an LTF, relative to the standard Gaussian distribution. This is equivalent to showing the following: for any unit vectors $W^1, \ldots, W^k \in \mathbb{R}^n$ and any $\vec{\theta} = (\theta_1, \ldots, \theta_k) \in \mathbb{R}^k$, taking W to be the $k \times n$ matrix whose rows are $W^1, \ldots, W^k$ and taking $\mathcal{O}$ to be any union of orthants over $\mathbb{R}^k$, we have

$$\left| \Pr_{\mathbf{Z} \leftarrow \mathsf{Gen}} [W\mathbf{Z} - \vec{\theta} \in \mathcal{O}] - \Pr_{\mathbf{G}^{(n)} \leftarrow \mathcal{N}(0,1)^n} [W\mathbf{G}^{(n)} - \vec{\theta} \in \mathcal{O}] \right| \leq \varepsilon. \tag{2}$$

Here is a high-level sketch of why our PRG ensures this.

(1) A (pseudorandom) JL projection of the k unit vectors $W^1, \ldots, W^k \in \mathbb{R}^n$ results in much lower-dimensional vectors $V^1, \ldots, V^k \in \mathbb{R}^d$, where $d = \Theta(\log(k)/\varepsilon'^2)$, which approximately preserve pairwise distances. Let us write Σ^W (Σ^V respectively) to denote the $k \times k$ covariance matrix of the k-dimensional Gaussian random variable $W\mathbf{G}^{(n)} - \vec{\theta}$ ($V\mathbf{G}^{(d)} - \vec{\theta}$ respectively, where $\mathbf{G}^{(d)}$ is distributed according to $\mathcal{N}(0,1)^d$). As we will see in Section 4.1, we have that Σ^W and Σ^V are entrywise close to each other (see Observation 6 for details).

(2) The entrywise closeness of Σ^W and Σ^V implies that the quadratic Wasserstein distance $\mathcal{W}_2(W\mathbf{G}^{(n)} - \vec{\theta}, V\mathbf{G}^{(d)} - \vec{\theta})$ is small; more precisely, we get that

$$\mathcal{W}_2(W\mathbf{G}^{(n)} - \vec{\theta}, V\mathbf{G}^{(d)} - \vec{\theta}) \leq \tau, \quad \text{where } \tau = O(k^{\frac{7}{8}} \cdot (\varepsilon')^{1/4}). \tag{3}$$

(See Proposition 8 in Section 5.2 for details.)

(3) As the main step of our analysis, using an adaptation of an argument from [5], in Section 5.3 we use (3) to infer that for every union of orthants $\mathcal{O}$, we have

$$\left| \Pr_{\mathbf{G}^{(n)} \leftarrow \mathcal{N}(0,1)^n} [W\mathbf{G}^{(n)} - \vec{\theta} \in \mathcal{O}] - \Pr_{\mathbf{G}^{(d)} \leftarrow \mathcal{N}(0,1)^d} [V\mathbf{G}^{(d)} - \vec{\theta} \in \mathcal{O}] \right| \leq O(k^{2/3}\tau^{2/3}) = \varepsilon. \tag{4}$$

This concludes the analysis since the inequality (4) is exactly the same as (2). This is because for each j we have $V^j = W^j\mathbf{A}^\mathsf{T}$ where $\mathbf{A}$ is the (pseudorandom) projection matrix.

[2] By the Kantorovich-Rubinstein duality theorem, there is an equivalent formulation $\mathcal{W}_2(\mathbf{X}, \mathbf{Y})$ in terms of Lipschitz test functions, but we will not need this alternative formulation.

2.2 An outline of our analysis for fooling intersections of LTFs

At a high level, our analysis for fooling intersections of LTFs exploits the rich and influential line of work on analyzing supremum (maximum) of Gaussian processes [24, 11, 31]. We recall that a Gaussian process is a set of jointly normal random variables (the set may be infinite, though we will only concerned with the finite case where it has cardinality k). To see the relationship between the maximum of a Gaussian process and an intersection of LTFs, let $W^1, \ldots, W^k \in \mathbb{R}^n$ be unit vectors and $\vec{\theta} \in \mathbb{R}^k$. Define the LTF $h_i(z) = \mathrm{sign}(W^i z - \theta_i)$ and consider the k-face polytope $h_1(z) \wedge \ldots \wedge h_k(z)$. Showing that our PRG ε-fools this k-face polytope (i.e., the function $h_1 \wedge \ldots \wedge h_k$) relative to the standard Gaussian distribution is equivalent to showing the following: Taking W to be the $k \times n$ matrix whose rows are $W^1, \ldots, W^k$,

$$\left| \Pr_{\mathbf{Z} \leftarrow \mathsf{Gen}} [W\mathbf{Z} \leq \vec{\theta}] - \Pr_{\mathbf{G} \leftarrow \mathcal{N}(0,1)^n} [W\mathbf{G} \leq \vec{\theta}] \right| \leq \varepsilon. \tag{5}$$

Note that $W\mathbf{Z} \leq \vec{\theta}$ if and only if $\max_{j \in [k]}((W\mathbf{Z})_j - \theta_j) \leq 0$. Likewise, $W\mathbf{G} \leq \vec{\theta}$ if and only if $\max_{j \in [k]}((W\mathbf{G})_j - \theta_j) \leq 0$.

Both $\{(W\mathbf{Z})_j - \theta_j\}_{1 \leq j \leq k}$ and $\{(W\mathbf{G})_j - \theta_j\}_{1 \leq j \leq k}$ are Gaussian processes, and we are interested in comparing the maxima of these two processes. If we were interested in comparing just the expectations of the maxima, i.e., $\mathbf{E}[\max_{j \in [k]}((W\mathbf{Z})_j - \theta_j)]$ versus $\mathbf{E}[\max_{j \in [k]}((W\mathbf{G})_j - \theta_j)]$, then the classical Sudakov-Fernique inequality [11, 30] provides a tool to compare (and prove the closeness of) these two quantities. Indeed, Meka [25] used this as a starting point in his work on a deterministic algorithm for estimating the supremum of a Gaussian process. We are interested in a somewhat more delicate quantity, and so we will use a generalization of a recent result of Chernozhukov *et al.* [6] which itself extends the Sudakov-Fernique inequality.

Now we turn from the above conceptual overview to a more detailed sketch of our analysis. Let the vectors $V^1, \ldots, V^k$ and the covariance matrix Σ^V be defined in the previous subsection.

(1') The first step of the argument is identical to Step 1 in the previous subsection: the covariance matrices Σ^W and Σ^V are entrywise close to each other.

(2') Next, we use the entrywise closeness of Σ^W and Σ^V to show that for any sufficiently smooth function g, we have that

$$\left| \mathbf{E}[g(\max_{j \in [k]} (W^j \cdot \mathbf{G}^{(n)} - \theta_j))] - \mathbf{E}[g(\max_{j \in [k]} (V^j \cdot \mathbf{G}^{(d)} - \theta_j))] \right| \text{ is small.} \tag{6}$$

is small. This is via an extension (to non-centered Gaussians) of Theorem 1 of [6], which in turn is a generalization of Chatterjee's quantitative Fernique-Sudakov bound [4].[3] We carry out this step in Section 6.2.

(3') Using a result of [16] (which follows almost directly from an influential work of Nazarov [27]), we have that the real-valued random variable

$$\max_{j \in [k]} (W^j \cdot \mathbf{G}^{(n)} - \theta_j),$$

which is a max of non-centered Gaussians, has good *anticoncentration*, meaning that it does not put very much mass in any small interval. See Section 6.3 for more details.

[3] Chatterjee's original argument in [4] bounds the difference in the expectations of $\max_{j \in [k]} (W^j \cdot \mathbf{G}^{(n)} - \theta_j)$ and $\max_{j \in [k]} (V^j \cdot \mathbf{G}^{(n)} - \theta_j)$, corresponding to the identity function $g(x) = x$.

(4') We specialize (6) to the case where g is a smooth approximator of the sign function. For a particular such g, combining (6) with the anticoncentration of $\max_{j\in[k]}(W^j\cdot\mathbf{G}^{(n)}-\theta_j)$ mentioned above, we can pass from g, which is a smooth approximator of sign$(\cdot)$, to the actual sign$(\cdot)$ function, and thereby show that

$$\left|\mathbf{Pr}[\mathrm{sign}(\max_{j\in[k]}(W^j\cdot\mathbf{G}^{(n)}-\theta_j))=1]-\mathbf{Pr}[\mathrm{sign}(\max_{j\in[k]}(V^j\cdot\mathbf{G}^{(d)}-\theta_j))=1]\right| \tag{7}$$

is small. We give this argument in Section 6.4.

(5') Having (7) be small is exactly the same as having the LHS of (5) is small, since for each j we have $V^j=W^j\mathbf{A}^\mathsf{T}$ where $\mathbf{A}$ is the (pseudorandom) projection matrix from Step 1 of our PRG. See Section 6.5 for more details.

3 The idea of our deterministic approximate counting results

In this section, we give an overview of our approximate counting algorithms for intersections and arbitrary functions of LTFs. We begin with the description for arbitrary functions as it relies on (extensions of) relatively well known tools from the literature such as regularity lemmas and invariance principles. In particular, we follow the (by now standard) paradigm of *reducing* the counting problem over the discrete cube to the Gaussian case by applying an appropriate regularity lemma; the proof of correctness relies on an *invariance principle for arbitrary functions of LTFs*. Once in the Gaussian case, we apply Theorem 3 which allows us to do counting over Gaussian space. This is explained in more detail in Section 3.1.

We then move on to the case of intersections of LTFs, which is somewhat more subtle. Similar to the first case, we also use a regularity lemma to *reduce* the Boolean case to the Gaussian case. However, instead of a naive approach of traversing all the root-to-leaf paths in the decision tree (constructed by the regularity lemma), we use the PRG construction of [28] to traverse only a small subset of the leaves. More details are given in Section 3.2.

3.1 Deterministic approximate counting for arbitrary functions of k LTFs via an invariance principle and a multiregularity lemma

A *regular* LTF is an LTF $\mathrm{sign}(\sum_{i=1}^n w_ix_i-\theta)$ in which, intuitively, no individual weight w_i has large magnitude compared to the overall magnitude of the weights (see Section 7.1 for a precise definition). The main structural result of [16] is an *invariance principle for intersections of LTFs*: roughly speaking, this states that if $F_0=h_1\wedge\cdots\wedge h_k$ is an intersection of k LTFs all of which are sufficiently regular, then the expected values of $F_0(\mathbf{U}^{(n)})$ (where the input is uniform over $\{-1,1\}^n$) and of $F_0(\mathbf{G}^{(n)})$ (where the input is a standard $\mathcal{N}(0,1)^n$ Gaussian) are close. A notable aspect of the [16] invariance principle is that its error bound has only a *poly-logarithmic* dependence on k (see Theorem 28 in Section 7.3 for a precise statement).

Now, consider any $F=g(h_1,\cdots,h_k)$ (where $g:\{-1,1\}^k\to\{-1,1\}$ is arbitrary). A naive approach based on just using the [16] invariance principle 2^k times together with a union bound would give an invariance principle for arbitrary functions of k LTFs with an error bound that depends exponentially on k. Instead, we use an analogue of the [16] invariance principle which goes beyond intersections of LTFs and works for arbitrary functions of k LTFs. The work of Gopalan et al. [14] gives an invariance principle for arbitrary functions of k LTFs that has a polynomial dependence on k in the error bound. We provide an alternate proof of this invariance principle for arbitrary functions of k LTFs. This polynomial dependence on k is crucial for obtaining a final overall running time for counting satisfying assignments with a singly exponential dependence on k, rather than a doubly exponential dependence which would follow from the naive approach.

As we explain in Section 7.2, the proof of our invariance principle is completely different from the proofs of of [16], [14]; we feel that our new proof of the invariance principle, Theorem 24, may be of independent interest. The [16] and [14] invariance principles are proved using a Lindeberg-type "replacement" argument; key ingredients are an analysis of hashing n coordinates into buckets and bounds on the derivatives of particular "smooth mollifiers" for functions of LTFs. Our proof of Theorem 24 uses none of these ingredients; instead, its main components are (a) a CLT for Wasserstein distance due to Valiant and Valiant [32], and (b) a conversion from Wasserstein distance to "union-of-orthants" distance. (Indeed, the ideas underlying the proof of Theorem 24 are very similar to the ideas underlying our PRG for arbitrary functions of k LTFs; this is analogous, at a high level, to how the proof of the [16] invariance principle is closely related to the analysis of the [16] PRG for intersections of regular LTFs.)

Using the invariance principle for deterministic approximate counting. By combining the invariance principle for arbitrary functions of LTFs with our PRG, which shows that a random variable $\mathbf{Z} \leftarrow \mathsf{Gen}$ is such that the expectation of $F(\mathbf{Z})$ is close to that of $F(\mathbf{G}^{(n)})$, it is straightforward to obtain a deterministic approximate counting algorithm for arbitrary functions of k regular LTFs over $\{-1,1\}^n$ simply by enumerating over all the seeds of our PRG. This algorithm has running time $\mathrm{poly}(n) \cdot 2^{\mathrm{poly}(k,1/\varepsilon)}$.) To obtain a deterministic approximate counting algorithm for arbitrary functions of k *general* LTFs over $\{-1,1\}^n$, we combine the above algorithm with the deterministic algorithmic version of the *multi-regularity lemma* of [14]. Briefly, this is a deterministic algorithm which builds a decision tree of depth roughly k, with the property that at almost every leaf ρ of the decision tree, either the restriction of $g(h_1, \cdots, h_k)$ according to ρ is very close to a constant function -1 or 1, or else each restricted LTF $h_1 \restriction \rho, \ldots h_k \restriction \rho$ is regular (and hence the deterministic approximate counting algorithm for arbitrary functions of regular LTFs can be used). We note that the total number of leaves in this decision tree is exponential in k. By running the approximate counting algorithm for functions of k-regular LTFs at each of the leaves, it is possible to approximate the overall number of satisfying assignments. We give the details of this (fairly standard) approach in Section 7.2.

3.2 Deterministic approximate counting for intersections of k LTFs

Let $F = h_1 \wedge \cdots \wedge h_k$. Recall that the invariance principle of [16] shows that if all the LTFs are sufficiently regular, then the expected values of $F_0(\mathbf{U}^{(n)})$ and of $F_0(\mathbf{G}^{(n)})$ are close, where crucially the error bound only has a polylogarithmic dependence on k. By combining this with our PRG, it is straightforward to obtain a deterministic approximate counting algorithm for intersections of k regular LTFs over $\{-1,1\}^n$ simply by enumerating over all the seeds of our PRG – the resulting running time is $\mathrm{poly}(n) \cdot 2^{\mathrm{poly}(\log k, 1/\varepsilon)}$. For intersections of general halfspaces, one can apply the multi-regularity lemma of [15] to reduce to the case of intersection of regular halfspaces. A naive application of this (similar to the previous subsection) will result in a running time exponential in k – this is because there are 2^k leaves in the resulting decision tree and running the algorithm for each of the leaves separately will result in an exponential in k overhead.

To instead get a $2^{\mathrm{poly}(\log k)}$ overhead, we crucially rely on two facts: (i) the decision tree constructed by the regularity lemma is non-adaptive, i.e., all nodes at the same level are labeled by the same variable. Further, if the set of internal variables is denoted by S, then this set can be enumerated in time $\mathrm{poly}(S)$. (ii) For any fixing of the set of variables in $\overline{S}$, the computation of the decision tree can be represented as an intersection of k halfspaces.

Glossing over some subtleties, this suggests that instead of doing approximate counting for all the leaves in the decision tree, one can just perform this computation on a subset of the leaves given by the output of a PRG. In particular, we use the PRG due to [28] to select the subset. While the PRG in [28] has a $(\log n) \cdot \mathrm{poly}(\log k)$ seed length (where n is the ambient dimension), in this application 'n' is set to $|S|$ which has polynomial dependence on k (for constant error $\varepsilon > 0$). Putting this together, we obtain a deterministic algorithm for counting intersection of k arbitrary halfspaces with running time $\mathrm{poly}(n) \cdot 2^{\mathrm{poly}(\log k, 1/\varepsilon)}$. The full details are given in Section 7.3.

4 Notation and setup

We write $W \in \mathbb{R}^{k \times n}$ to denote the matrix whose j-th row is the weight vector of the j-th LTF in a function of k LTFs. We assume that each such LTF has been normalized so that its weight vector has norm 1. For $j \in [k]$ (indexing one of the LTFs) we write $W^j = (W^j_1, \ldots, W^j_n)$ to denote the j-th row of W, so $\|W^j\| = 1$ for all j. Thus an arbitrary function of k LTFs is $g(h_1, \ldots, h_k)$, where $g : \{-1,1\}^k \to \{-1,1\}$ and

$$h_j(x) = \mathrm{sign}(W^j \cdot x - \theta_j) \qquad \text{where } W^j = (W^j_1, \ldots, W^j_n) \in \mathbb{R}^n \text{ has } \|W^j\| = 1$$

(we take -1 to represent True and 1 to represent False throughout), and an intersection of k LTFs is a function $h_1(x) \wedge \cdots \wedge h_k(x)$.

Throughout this paper we will use notation like $\vec{\theta}$ to denote vectors in $\mathbb{R}^k$, i.e. $\vec{\theta} = (\theta_1, \ldots, \theta_k) \in \mathbb{R}^k$. We write $\mathbf{G}$ or simply $\mathbf{G}^{(n)}$ to denote $(\mathbf{G}_1, \ldots, \mathbf{G}_n)$, a random variable distributed according to $\mathcal{N}(0,1)^n$ (so each of $\mathbf{G}_1, \ldots, \mathbf{G}_n$ is an i.i.d. $\mathcal{N}(0,1)$ Gaussian).

4.1 Entrywise closeness of the original covariance matrix and the pseudorandomly-projected covariance matrix

As above let $W \in \mathbb{R}^{k \times n}$ have j-th row W^j with $\|W^j\| = 1$ for all $j \in [k]$. For convenience we also define $W^0 \in \mathbb{R}^n$ to be the all-0 vector.

Let $d = O(\log(k/\delta')/\varepsilon'^2)$ (where ε' will be taken to be at most 1) and let $V \in \mathbb{R}^{k \times d}$ satisfy the following:

$$\text{For all } 0 \le i, j \le k \text{ we have } \|W^i - W^j\| \le \|V^i - V^j\| \le (1+\varepsilon')\|W^i - W^j\| \tag{8}$$

where we take $V^0 = (0, \ldots, 0) \in \mathbb{R}^d$. (As we will see in the next subsection, $V^1, \ldots, V^k$ should be thought of as the vectors we get by doing a pseudorandom JL-projection of $W^1, \ldots, W^k$ to d dimensions.)

We will consider the two k-dimensional Gaussian random vectors $W\mathbf{G}^{(n)}$ and $V\mathbf{G}^{(d)}$. The covariance matrix of $W\mathbf{G}^{(n)}$, which we denote Σ^W, is the $k \times k$ matrix $W^{\mathsf{T}}W$ which has $\sigma^W_{ij} := W^i \cdot W^j$ as its (i,j) entry, and similarly the covariance matrix Σ^V of $V\mathbf{G}^{(d)}$ has $\sigma^V_{ij} := V^i \cdot V^j$ as its (i,j) entry. We define

$$\Delta := \max_{1 \le i,j \le k} |\sigma^W_{ij} - \sigma^V_{ij}| = \max_{1 \le i,j \le k} |W^i \cdot W^j - V^i \cdot V^j|, \tag{9}$$

the maximum entry-wise difference between the two covariance matrices. The following simple observation upper bounds Δ:

▶ **Observation 6.** *If $W^0, \ldots, W^k \in \mathbb{R}^n$, $V^0, \ldots, V^k \in \mathbb{R}^d$ satisfy (8), then $\Delta \le 9\varepsilon'$.*

Proof. Taking $i = 0$, (8) implies that each V^j, $j \in [k]$, has $\|V^j\| \in [1, 1+\varepsilon']$. Now fix any $i, j \in [k]$. We have

$$\|W^i - W^j\|^2 = W^i \cdot W^i - 2W^i \cdot W^j + W^j \cdot W^j = 2 - 2W^i \cdot W^j$$

and similarly (using the fact that each $\|V^\ell\|^2 \le (1+\varepsilon')^2$)

$$\|V^i - V^j\|^2 = V^i \cdot V^i - 2V^i \cdot V^j + V^j \cdot V^j = 2 + 2\gamma - 2V^i \cdot V^j$$

for some $0 \le \gamma \in 2\varepsilon' + \varepsilon'^2 \le 3\varepsilon'$. Hence

$$2\gamma + 2W^i \cdot W^j - 2V^i \cdot V^j = \|V^i - V^j\|^2 - \|W^i - W^j\|^2,$$

which implies

$$\begin{aligned}
|W^i \cdot W^j - V^i \cdot V^j| &\le \gamma + \frac{1}{2}\left(\|V^i - V^j\|^2 - \|W^i - W^j\|^2\right) \\
&\le 3\varepsilon' + \frac{1}{2}\left(\left((1+\varepsilon')\|W^i - W^j\|\right)^2 - \|W^i - W^j\|^2\right) \\
&= 3\varepsilon' + \frac{1}{2}\left((2\varepsilon' + \varepsilon'^2)\|W^i - W^j\|^2\right) \\
&\le 3\varepsilon' + 2(2\varepsilon' + \varepsilon'^2) \le 9\varepsilon',
\end{aligned}$$

where for the penultimate inequality we used $\|W^i - W^j\|^2 \le 4$ and $\varepsilon'^2 \le \varepsilon'$ which holds since $0 < \varepsilon' < 1$. ◀

4.2 Formalizing step (1) of the intuitive sketch: Getting d-dimensional vectors $V^1, \ldots, V^k$ via pseudorandom projection

Recall that Steps 1 and 1′ of the analysis are identical for arbitrary functions of LTFs (in Section 2.1) and for intersections of LTFs (in Section 2.2). We give the details of this step here.

We use the following derandomized JL lemma given by Kane, Meka, and Nelson [20]:

▶ **Theorem 7** (Derandomized Johnson-Lindenstrauss [20]). *Let $0 \le \varepsilon', \delta' < 1/2$ and let $\delta'' = \delta'/k^2$. There is a distribution $\mathcal{D}$ over random matrices $\mathbf{A} \in \mathbb{R}^{d \times n}$, $d = O(\log(k/\delta')/\varepsilon'^2)$, such that (i) a draw of $\mathbf{A} \leftarrow \mathcal{D}$ can be generated using $O(\log n + \log(1/\delta'') \cdot \log((\log(1/\delta''))/\varepsilon'))$ bits, and (ii) the following holds: Fix unit vectors $W^1, \ldots, W^k \in \mathbb{R}^n$. Then*

$$\Pr_{\mathbf{A} \leftarrow \mathcal{D}}\left[\|W^i - W^j\| \le \|W^i\mathbf{A}^\mathsf{T} - W^j\mathbf{A}^\mathsf{T}\| \le (1+\varepsilon')\|W^i - W^j\| \text{ for all } i, j \in [k]\right] \ge 1 - \delta'. \tag{10}$$

Let $\mathbf{V}^j = W^j\mathbf{A}^\mathsf{T}$ where $\mathbf{A} \leftarrow \mathcal{D}$. By Theorem 7, except with failure probability at most δ', (8) is satisfied. We will always take $\delta' = \varepsilon$, and so this δ' failure probability just gets absorbed into the overall $O(\varepsilon)$ error bound of the PRG. Fix $V^1, \ldots, V^k$ to be any such outcome of $\mathbf{V}^1, \ldots, \mathbf{V}^k$; in the rest of the argument we will work with this $V^1, \ldots, V^k$. Note that by Observation 6 we have that Δ, which is defined in terms of this $V^1, \ldots, V^k$, satisfies $\Delta \le 9\varepsilon'$.

5 Fooling arbitrary functions of LTFs: Proof of Theorem 2

5.1 Parameter settings

As will be seen in the analysis below, in order for the overall PRG to $O(\varepsilon)$-fool arbitrary functions of k LTFs, we take $\varepsilon' = \frac{\varepsilon^6}{k^{15/2}}$. Recalling that $\delta' = \varepsilon$, by (1) the overall seed length (as a function of n, k and ε) is $O(\log n) + \tilde{O}(\frac{k^{15}}{\varepsilon^{12}})$, as claimed in Theorem 2. In the rest of this section we establish correctness of the PRG.

5.2 Formalizing step (2) of the intuitive sketch: Upper bounding the quadratic Wasserstein distance

Recall that the *quadratic Wasserstein distance* between random variables $\mathbf{X}, \mathbf{Y}$ in $\mathbb{R}^k$ is defined to be

$$\mathcal{W}_2(\mathbf{X}, \mathbf{Y}) = \inf_{(\widehat{\mathbf{X}}, \widehat{\mathbf{Y}})} (\mathbf{E}[\|\widehat{\mathbf{X}} - \widehat{\mathbf{Y}}\|^2])^{1/2}, \tag{11}$$

where the infimum is taken over all couplings $(\widehat{\mathbf{X}}, \widehat{\mathbf{Y}})$ of $\mathbf{X}$ and $\mathbf{Y}$.

▶ **Proposition 8.** *Let $W^1, \ldots, W^k$ be unit vectors in $\mathbb{R}^n$, $V^1, \ldots, V^k$ be vectors in $\mathbb{R}^d$ satisfying (8) and let $\vec{\theta} \in \mathbb{R}^k$. Then we have*

$$\mathcal{W}_2(W\mathbf{G}^{(n)} - \vec{\theta}, V\mathbf{G}^{(d)} - \vec{\theta}) \leq \tau, \qquad \text{where } \tau = O(k^{\frac{7}{8}} \cdot (\varepsilon')^{1/4}). \tag{12}$$

Proof. Observe that $W\mathbf{G}^{(n)} - \vec{\theta}$ and $V\mathbf{G}^{(d)} - \vec{\theta}$ have the same mean. For this case, Proposition 7 of Givens and Shortt [12] shows that

$$\mathcal{W}_2^2(W\mathbf{G}^{(n)} - \vec{\theta}, V\mathbf{G}^{(d)} - \vec{\theta}) = \mathrm{Tr}(\Sigma^W + \Sigma^V - 2((\Sigma^W)^{1/2}\Sigma^V(\Sigma^W)^{1/2})^{1/2}). \tag{13}$$

Here Σ^W and Σ^V are the covariance matrices of the distribution $W\mathbf{G}^{(n)} - \vec{\theta}$ and $V\mathbf{G}^{(d)} - \vec{\theta}$ respectively[4]. To bound the expression on the right hand side, first observe that

$$\left|\mathrm{Tr}(\Sigma^W + \Sigma^V) - 2\mathrm{Tr}(\Sigma^W)\right| \leq \left|\mathrm{Tr}(\Sigma^W - \Sigma^V)\right| \leq 9k \cdot \varepsilon'. \tag{14}$$

The last inequality uses Observation 6. To proceed further, we recall the following very useful fact from Bhatia [2] (Theorem X.1.3)

▶ **Fact 9.** *Let $\|\cdot\|$ be any unitarily invariant matrix norm. For psd matrices A and B, we have the following*

$$\| \, |A^{\frac{1}{2}} - B^{\frac{1}{2}}| \, \| \leq \|\sqrt{|A - B|}\|,$$

where $|X|$ denotes the psd matrix $\sqrt{X^*X}$.

For any symmetric matrix X, let $\|X\|_{\mathsf{tr}}$ denotes its trace norm, i.e., the sum of the singular values of X. Note that the trace-norm is unitarily invariant. With this, we now have

$$\begin{aligned} \left|2\mathrm{Tr}(\Sigma^W - ((\Sigma^W)^{1/2}\Sigma^V(\Sigma^W)^{1/2})^{1/2})\right| &\leq 2\|\Sigma^W - ((\Sigma^W)^{1/2}\Sigma^V(\Sigma^W)^{1/2})^{1/2}\|_{\mathsf{tr}} \\ &\leq 2\|\sqrt{|(\Sigma^W)^2 - (\Sigma^W)^{1/2}\Sigma^V(\Sigma^W)^{1/2}|}\|_{\mathsf{tr}} \\ &= 2\|\sqrt{|(\Sigma^W)^{1/2}(\Sigma^W - \Sigma^V)(\Sigma^W)^{1/2}|}\|_{\mathsf{tr}} \end{aligned} \tag{15}$$

In the above, the first inequality uses the fact that for any symmetric matrix X, $|\mathrm{Tr}(X)| \leq \|X\|_{\mathsf{tr}}$ and the second inequality follows from Fact 9. We now recall the following fact:

▶ **Fact 10.** *For any symmetric $X \in \mathbb{R}^{k \times k}$,*

$$\|\sqrt{|X|}\|_{\mathsf{tr}} \leq \sqrt{k} \cdot \sqrt{\|X\|_{\mathsf{tr}}}.$$

[4] [12] states their theorem for non-singular Σ^V and Σ^W. However, we can always perturb our Gaussians infinitesimally, apply (13) and then take a limit.

Proof. If $\sigma_1, \ldots, \sigma_k$ denotes the singular values of X, then the left hand side is $\sum_{j=1}^{k} \sqrt{\sigma_j}$ and the right hand side is $\sqrt{k} \cdot \sqrt{\sigma_1 + \ldots + \sigma_k}$, so the inequality is a consequence of the AM-GM inequality. ◀

Applying Fact 10 to (15), we have that

$$\begin{aligned} \left|2\mathrm{Tr}(\Sigma^W - ((\Sigma^W)^{1/2}\Sigma^V(\Sigma^W)^{1/2})^{1/2})\right| &\leq 2\sqrt{k}\sqrt{\||(\Sigma^W)^{1/2}(\Sigma^W - \Sigma^V)(\Sigma^W)^{1/2}|\|_{\mathsf{tr}}}. \\ &= 2\sqrt{k}\sqrt{\|(\Sigma^W)^{1/2}(\Sigma^W - \Sigma^V)(\Sigma^W)^{1/2}\|_{\mathsf{tr}}}. \quad (16) \end{aligned}$$

The second equality simply uses that for symmetric X, $\||X|\|_{\mathsf{tr}} = \|X\|_{\mathsf{tr}}$. Next, we recall the following useful inequality for unitarily invariant norms (see [2], p.94).

▶ **Fact 11.** *Let A, B, C be symmetric matrices and let $\|\cdot\|$ be any unitarily invariant norm. Then, $\|ABC\| \leq \|A\|_2 \cdot \|B\| \cdot \|C\|_2$.*

Applying Fact 11 to the right hand side of (16), we obtain

$$\begin{aligned} \left|2\mathrm{Tr}(\Sigma^W - ((\Sigma^W)^{1/2}\Sigma^V(\Sigma^W)^{1/2})^{1/2})\right| &\leq 2\sqrt{k}\sqrt{\|(\Sigma^W)^{1/2}\|_2\|\Sigma^W - \Sigma^V\|_{\mathsf{tr}}\|(\Sigma^W)^{1/2}\|_2}. \\ &= 2\sqrt{k}\|(\Sigma^W)^{1/2}\|_2 \cdot \sqrt{\|\Sigma^W - \Sigma^V\|_{\mathsf{tr}}}. \quad (17) \end{aligned}$$

Now, Σ^W is a matrix in which each entry $W^i \cdot W^j$ is upper bounded by 1 in absolute value. Thus, $\|\Sigma^W\|_2 \leq k$. This immediately implies that $\|(\Sigma^W)^{1/2}\|_2 \leq \sqrt{k}$. Similarly,

$$\|\Sigma^W - \Sigma^V\|_{\mathsf{tr}} \leq \sqrt{k} \cdot \|\Sigma^W - \Sigma^V\|_F \leq 9\sqrt{k} \cdot k \cdot \varepsilon' = 9\varepsilon' \cdot k^{3/2}.$$

Here the last inequality is again using Observation 6. Combining this with (17), we have

$$\left|2\mathrm{Tr}(\Sigma^W - ((\Sigma^W)^{1/2}\Sigma^V(\Sigma^W)^{1/2})^{1/2})\right| \leq 6k^{\frac{7}{4}} \cdot \sqrt{\varepsilon'}.$$

Combining the above equation with (14) and (13) (and using triangle inequality), we get that

$$\mathcal{W}_2^2(W\mathbf{G}^{(n)} - \vec{\theta}, V\mathbf{G}^{(d)} - \vec{\theta}) \leq 9k\varepsilon' + 2k^{\frac{7}{4}} \cdot \sqrt{\varepsilon'}.$$

This immediately yields the proposition. ◀

5.3 Formalizing step (3) of the intuitive sketch: Upper bounding the "union-of-orthants distance"

The following definition will be convenient: Given two random variables $\mathbf{X}, \mathbf{Y}$ over $\mathbb{R}^k$, the *union-of-orthants distance between* $\mathbf{X}$ *and* $\mathbf{Y}$ is defined to be

$$d_{\mathrm{UO}}(\mathbf{X}, \mathbf{Y}) := \max_{\mathcal{O}} \left|\mathbf{Pr}[\mathbf{X} \in \mathcal{O}] - \mathbf{Pr}[\mathbf{Y} \in \mathcal{O}]\right|, \quad (18)$$

where the max is taken over all 2^{2^k} possible unions of orthants $\mathcal{O}$ in $\mathbb{R}^k$. This definition aligns well with arbitrary functions of k LTFs $g(h_1, \ldots, h_k)$ because of the following easy observation:

▶ **Observation 12.** *For any $g : \{-1,1\}^k \to \{-1,1\}$ and any random variables $\mathbf{X}, \mathbf{Y}$ over $\mathbb{R}^k$, we have*

$$|\boldsymbol{Pr}[g(\mathrm{sign}(\mathbf{X}_1), \ldots, \mathrm{sign}(\mathbf{X}_k)) = 1] - \boldsymbol{Pr}[g(\mathrm{sign}(\mathbf{Y}_1), \ldots, \mathrm{sign}(\mathbf{Y}_k)) = 1]| \leq d_{\mathrm{UO}}(\mathbf{X}, \mathbf{Y}).$$

▶ **Lemma 13.** *Let $W^1, \ldots, W^k$ be unit vectors in $\mathbb{R}^n$, $V^1, \ldots, V^k$ be vectors in $\mathbb{R}^d$ satisfying (8) and let $\vec{\theta} \in \mathbb{R}^k$. Then we have*

$$d_{\mathrm{UO}}(W\mathbf{G}^{(n)} - \vec{\theta}, V\mathbf{G}^{(d)} - \vec{\theta}) \leq O(k^{2/3}\tau^{2/3}), \tag{19}$$

where τ is as defined in Proposition 8.

The argument here is similar to the proof of Theorem 5 in [5]. That result used a CLT due to Valiant and Valiant (which gave an upper bound on the L^1 (as opposed to quadratic, i.e. $\mathcal{W}_2$) transportation distance between a certain sum of vector-valued random variables and a Gaussian distribution) to obtain an upper bound on union-of-orthants distance between those two distributions. We briefly explain the main idea (which is quite simple) behind the argument in our setting.

We consider an optimal coupling of the random variables $\mathbf{X} = W\mathbf{G}^{(n)} - \vec{\theta}$ and $\mathbf{Y} = V\mathbf{G}^{(d)} - \vec{\theta}$ which achieves the minimal quadratic transportation distance as in (11). Since by Proposition 8 the quadratic transportation cost $\mathcal{W}_2(\mathbf{X}, \mathbf{Y})$ of transforming $\mathbf{X}$ to $\mathbf{Y}$ is "small", the optimal coupling cannot move a "non-small" amount of mass by a distance that is not "small." Assume (contrary to our desired conclusion) that the union-of-orthants distance between $\mathbf{X}$ and $\mathbf{Y}$ is not small, and fix a union of orthants $\mathcal{O}$ that achieves the max in (18). Without loss of generality we may suppose that $\mathbf{X}$ puts more mass on $\mathcal{O}$ than $\mathbf{Y}$ (and this difference is large by the above assumption). Gaussian anticoncentration tells us that $\mathbf{X}$ can only have a small amount of mass overall that is close to orthant boundaries, and hence $\mathbf{X}$ can have only a small amount of such mass in $\mathcal{O}$. This means that a non-small amount of mass from $\mathbf{X}$ must be moved a non-small distance (since it must go from being within $\mathcal{O}$ and not close to any orthant boundary, to being outside of $\mathcal{O}$) in order to transform $\mathbf{X}$ to $\mathbf{Y}$; but this contradicts the premise that $\mathcal{W}_2(\mathbf{X}, \mathbf{Y})$ is small.

We now proceed to the formal argument.

Proof of Lemma 13. As above let $\mathbf{X} = W\mathbf{G}^{(n)} - \vec{\theta}$ and $\mathbf{Y} = V\mathbf{G}^{(d)} - \vec{\theta}$. By Proposition 8 we have that $\mathcal{W}_2(\mathbf{X}, \mathbf{Y}) \leq \tau$. We define

$$B_r := \{x \in \mathbb{R}^k : |x_i| \leq r \text{ for some } i \in [k]\}$$

to be the region of all points in $\mathbb{R}^k$ whose L^∞-distance from any orthant boundary point is at most r. With foresight we choose $r = \tau^{2/3}/k^{1/3}$ (the rationale for this choice will be evident toward the end of the proof). We partition $\mathcal{O}$ into $\mathcal{O}_{\mathrm{bd}} := \mathcal{O} \cap B_r$ (the points in $\mathcal{O}$ that lie close to the orthant boundaries) and $\mathcal{O}_{\mathrm{in}} := \mathcal{O} \setminus B_r$ (the points in $\mathcal{O}$ that lie far away from the orthant boundaries). We have

$$\begin{aligned} \big|\mathbf{Pr}[\mathbf{X} \in \mathcal{O}] - \mathbf{Pr}[\mathbf{Y} \in \mathcal{O}]\big| &= \big|(\mathbf{Pr}[\mathbf{X} \in \mathcal{O}_{\mathrm{in}}] + \mathbf{Pr}[\mathbf{X} \in \mathcal{O}_{\mathrm{bd}}]) - (\mathbf{Pr}[\mathbf{Y} \in \mathcal{O}_{\mathrm{in}}] + \mathbf{Pr}[\mathbf{Y} \in \mathcal{O}_{\mathrm{bd}}])\big| \\ &\leq \underbrace{\big|\mathbf{Pr}[\mathbf{X} \in \mathcal{O}_{\mathrm{in}}] - \mathbf{Pr}[\mathbf{Y} \in \mathcal{O}_{\mathrm{in}}]\big|}_{=\Xi} + \underbrace{\mathbf{Pr}[\mathbf{X} \in \mathcal{O}_{\mathrm{bd}}] + \mathbf{Pr}[\mathbf{Y} \in \mathcal{O}_{\mathrm{bd}}]}_{=\Gamma}. \end{aligned}$$

We bound the quantities Ξ and Γ separately.

For Γ, we have that

$$\Gamma \leq \sum_{i=1}^{k} \mathbf{Pr}\left[\mathbf{X}_i \in [-r, r]\right] + \mathbf{Pr}\left[\mathbf{Y}_i \in [-r, r]\right] \leq O(kr), \tag{20}$$

where we used the fact that each coordinate $\mathbf{X}_i$ of $\mathbf{X}$ is a one-dimensional Gaussian with variance $\|W^i\|^2 = 1$ and each coordinate $\mathbf{Y}_i$ of $\mathbf{Y}$ is a one-dimensional Gaussian with variance $1 \leq \|V^i\|^2 \leq (1 + \varepsilon')^2 = O(1)$.

For Ξ, let us assume without loss of generality (a symmetrical argument works in the other case) that $\mathbf{Pr}[\mathbf{X} \in \mathcal{O}_{\text{in}}] \geq \mathbf{Pr}[\mathbf{Y} \in \mathcal{O}_{\text{in}}]$, so $\Xi = \mathbf{Pr}[\mathbf{X} \in \mathcal{O}_{\text{in}}] - \mathbf{Pr}[\mathbf{Y} \in \mathcal{O}_{\text{in}}]$. Let $\mathcal{D}$ be any coupling of $\mathbf{X}$ and $\mathbf{Y}$ that achieves

$$\mathop{\mathbf{E}}_{(\widehat{\mathbf{X}},\widehat{\mathbf{Y}})\sim\mathcal{D}}[\|\widehat{\mathbf{X}} - \widehat{\mathbf{Y}}\|^2]^{1/2} = 2\tau,$$

so $\mathcal{D}$ is the joint distribution of a pair $(\mathbf{U}, \mathbf{V})$ of $\mathbb{R}^k$-valued random variables with marginals distributed according to $\mathbf{X}$ and $\mathbf{Y}$ respectively. Since

$$\int_{\mathcal{O}_{\text{in}}} \int_{\mathbb{R}^k} \mathcal{D}(u,v)\, dv\, du = \mathbf{Pr}[\mathbf{X} \in \mathcal{O}_{\text{in}}]$$

and

$$\int_{\mathcal{O}_{\text{in}}} \int_{\mathcal{O}_{\text{in}}} \mathcal{D}(u,v)\, dv\, du \leq \int_{\mathbb{R}^k} \int_{\mathcal{O}_{\text{in}}} \mathcal{D}(u,v)\, dv\, du = \mathbf{Pr}[\mathbf{Y} \in \mathcal{O}_{\text{in}}],$$

it follows that

$$\int_{\mathcal{O}_{\text{in}}} \int_{\mathbb{R}^k \setminus \mathcal{O}_{\text{in}}} \mathcal{D}(u,v)\, dv\, du = \int_{\mathcal{O}_{\text{in}}} \int_{\mathbb{R}^k} \mathcal{D}(u,v)\, dv\, du - \int_{\mathcal{O}_{\text{in}}} \int_{\mathcal{O}_{\text{in}}} \mathcal{D}(u,v)\, dv\, du \geq \Xi. \tag{21}$$

Next we define the quantities

$$\Xi_{\text{near}}(\mathcal{D}) \quad := \quad \int_{\mathcal{O}_{\text{in}}} \int_{\mathcal{O}_{\text{bd}}} \mathcal{D}(u,v)\, dv\, du$$

(in words, this is the probability that $\mathbf{U}$ lies "well inside" $\mathcal{O}$ and $\mathbf{V}$ lies "close to the boundary" in $\mathcal{O}$), and

$$\Xi_{\text{far}}(\mathcal{D}) \quad := \quad \int_{\mathcal{O}_{\text{in}}} \int_{\mathbb{R}^k \setminus \mathcal{O}} \mathcal{D}(u,v)\, dv\, du$$

(in words, this is the probability that $\mathbf{U}$ lies "well inside" $\mathcal{O}$ and $\mathbf{V}$ lies outside $\mathcal{O}$). Note that $\Xi_{\text{near}}(\mathcal{D})$ and $\Xi_{\text{far}}(\mathcal{D})$ sum to the quantity on the left-hand side of (21), and so $\Xi_{\text{near}}(\mathcal{D}) + \Xi_{\text{far}}(\mathcal{D}) \geq \Xi$. (In words, since $\mathbf{X}$ places Ξ more mass on $\mathcal{O}_{\text{in}}$ than $\mathbf{Y}$ does, any scheme $\mathcal{D}$ of moving the mass of $\mathbf{X}$ to obtain $\mathbf{Y}$ must move at least Ξ amount from within $\mathcal{O}_{\text{in}}$ to outside it. $\Xi_{\text{near}}(\mathcal{D})$ is the amount moved from within $\mathcal{O}_{\text{in}}$ to $\mathcal{O}$'s boundary $\mathcal{O}_{\text{bd}}$, and $\Xi_{\text{far}}(\mathcal{D})$ is the rest, moved from within $\mathcal{O}_{\text{in}}$ to locations entirely out of $\mathcal{O}$.) Since $\|u - v\|^2 \geq r^2$ for any pair of points $u \in \mathcal{O}_{\text{in}}$ and $y \notin \mathcal{O}$, it follows that

$$(2\tau)^2 = \mathop{\mathbf{E}}_{(\mathbf{U},\mathbf{V})\sim\mathcal{D}}[\|\mathbf{U} - \mathbf{V}\|^2] \geq r^2 \cdot \Xi_{\text{far}}(\mathcal{D}).$$

We consider two cases, depending on the relative magnitudes of $\Xi_{\text{near}}(\mathcal{D})$ and $\Xi_{\text{far}}(\mathcal{D})$. If $\Xi_{\text{far}}(\mathcal{D}) \geq \Xi_{\text{near}}(\mathcal{D})$, then we have

$$r^2 \cdot \frac{\Xi}{2} \leq r^2 \cdot \Xi_{\text{far}}(\mathcal{D}) \leq 4\tau^2,$$

and hence $\Xi \leq 8\tau^2/r^2$, which along with our upper bound on Γ given by (20) completes the proof. If on the other hand $\Xi_{\text{near}}(\mathcal{D}) > \Xi_{\text{far}}(\mathcal{D})$, then

$$\frac{\Xi}{2} \leq \Xi_{\text{near}}(\mathcal{D}) \leq \int_{\mathbb{R}^k} \int_{\mathcal{O}_{\text{bd}}} \mathcal{D}(u,v)\, dv\, du = \mathbf{Pr}[\mathbf{Y} \in \mathcal{O}_{\text{bd}}] \leq \Gamma,$$

and again our upper bound on Γ completes the proof. ◀

Observing that by our setting of parameters we have that $k^{2/3}\tau^{2/3} = O(\varepsilon)$, we get that

$$d_{\mathrm{UO}}(W\mathbf{G}^{(n)} - \vec{\theta}, V\mathbf{G}^{(d)} - \vec{\theta}) \leq O(\varepsilon)$$

provided that $W^1, \ldots, W^k$, $V^1, \ldots, V^k$ satisfy (8). Recalling from Section 4.2 that all but a $\delta' = \varepsilon$ fraction of outcomes $V^1, \ldots, V^k$ of $\mathbf{V}^j = W^j\mathbf{A}^\mathsf{T}$ satisfy (8), we have

$$d_{\mathrm{UO}}(W\mathbf{G}^{(n)} - \vec{\theta}, W\mathbf{A}^\mathsf{T}\mathbf{G}^{(d)} - \vec{\theta}) \leq O(\varepsilon),$$

and recalling that a draw $\mathbf{Z}$ from our generator Gen is $\mathbf{Z} = \mathbf{A}^\mathsf{T}\mathbf{G}^{(d)}$, this is equivalent to

$$d_{\mathrm{UO}}(W\mathbf{G}^{(n)} - \vec{\theta}, W\mathbf{Z} - \vec{\theta}) \leq O(\varepsilon),$$

and the proof of Theorem 2 is complete.

6 Fooling intersections of LTFs: Proof of Theorem 3

6.1 Parameter settings, notation and terminology

As we will see in the analysis given below, in order for the overall PRG to ε-fool k-facet Gaussian polytopes it suffices to take $\varepsilon' = O(\varepsilon^3/\log^2 k)$ and $\delta' = \varepsilon'/k^2$, so by (1) the overall seed length (as a function of n, k and ε) is $O(\log n) + \tilde{O}(\frac{\log^6 k}{\varepsilon^6})$ as claimed in Theorem 3.

The following notation will be useful: For $0 < \lambda$, $k \geq 1$, and $\vec{\theta} = (\theta_1, \ldots, \theta_k) \in \mathbb{R}^k$, we define

$$\mathrm{Strip}_{\lambda,k,\vec{\theta}} = \{x \in \mathbb{R}^k : \text{ some } j \in [k] \text{ has } x_j \in (\theta_j, \theta_j + \lambda) \text{ and every } j \in [k] \text{ has } x_j < \theta_j + \lambda\}.$$

We recall that the *Kolmogorov distance* between two real-valued random variables $\mathbf{S}$ and $\mathbf{T}$ is defined to be

$$d_{\mathrm{K}}(\mathbf{S}, \mathbf{T}) = \sup_{\theta \in \mathbb{R}} \big|\, \mathbf{Pr}[\mathbf{S} \leq \theta] - \mathbf{Pr}[\mathbf{T} \leq \theta] \big|.$$

For $f : \mathbb{R}^k \to \mathbb{R}$ a smooth function we write $\partial_j f(z)$ to denote $\frac{\partial f}{\partial z_j}(z)$ and write $\partial_i \partial_j f(z)$ to denote $\frac{\partial^2 f}{\partial z_i \partial z_j}(z)$.

6.2 Formalizing step (2′) of the intuitive sketch: Fooling smooth test functions of max of non-centered Gaussians

A crucial ingredient in executing step (2′) of our analysis is the the following "soft-max" function which is used in [4, 6] and many other works. The soft-max function $F_\beta : \mathbb{R}^k \to \mathbb{R}$ is defined as

$$F_\beta(x_1, \ldots, x_k) = \frac{1}{\beta} \cdot \ln\left(\sum_{i=1}^{k} e^{\beta x_i}\right).$$

For conciseness let us write e_β to denote $\beta^{-1} \ln k$. We record some useful facts about the soft-max function:

▶ **Fact 14.** *For any vector $v \in \mathbb{R}^k$, and any parameter $\beta > 0$,*

$$0 \leq F_\beta(v) - \max_{i \in [k]} v_i \leq e_\beta.$$

▶ **Fact 15** (Lemma 3 of [6]). *For every $1 \le i, j \le k$, we have*

$$\partial_i F_\beta(z) = \pi_i(z), \qquad \partial_i \partial_j F_\beta(z) = \beta w_{ij}(z),$$

where

$$\pi_i(z) := \frac{e^{\beta z_i}}{\sum_{\ell=1}^k e^{\beta z_\ell}}, \qquad w_{ij}(z) := \mathbb{1}[i=j]\pi_i(z) - \pi_i(z)\pi_j(z).$$

Furthermore, we have

$$\pi_j(z) \ge 0, \qquad \sum_{j=1}^k \pi_j(z) = 1, \qquad \sum_{i=1}^k \sum_{j=1}^k |w_{ij}(z)| \le 2.$$

▶ **Fact 16** (Lemma 4 of [6]). *Let $m(z) = g(F_\beta(z))$ where $g \in C^2(\mathbb{R})$. Then for every $1 \le i, j \le k$, we have*

$$\partial_i \partial_j m(z) = (g''(F_\beta(z))\pi_i(z)\pi_j(z) + \beta g'(F_\beta(z)) w_{ij}(z),$$

where π_i and w_{ij} are defined as in Fact 15 above.

Fact 14 follows almost directly from the definition of F_β. Facts 15 and 16 can be routinely verified by calculus.

The following is the main result of this section (cf. (6)):

▶ **Theorem 17** (Fooling smooth test functions of max of non-centered Gaussians). *Let $W^1, \dots, W^k$ be unit vectors in $\mathbb{R}^n$, $V^1, \dots, V^k$ be vectors in $\mathbb{R}^d$ satisfying (8) and let $\vec{\theta} \in \mathbb{R}^k$. Fix any function $g \in C^2(\mathbb{R})$, $g : \mathbb{R} \to [-1, 1]$ such that $\|g'\|_\infty := \sup_{x \in \mathbb{R}} |g'(x)| < \infty$ and $\|g''\|_\infty := \sup_{x \in \mathbb{R}} |g''(x)| < \infty$. Then for any $\beta > 0$, we have*

$$\Big| \boldsymbol{E}[g(F_\beta(W^1 \cdot \mathbf{G}^{(n)} - \theta_1, \dots, W^k \cdot \mathbf{G}^{(n)} - \theta_k))] - \boldsymbol{E}[g(F_\beta(V^1 \cdot \mathbf{G}^{(d)} - \theta_1, \dots, V^k \cdot \mathbf{G}^{(d)} - \theta_k))] \Big| \le O(\|g''\|_\infty \varepsilon' + \|g'\|_\infty \varepsilon' \beta).$$

Further,

$$\left| \boldsymbol{E}[g(\max_{j \in [k]}(W^j \cdot \mathbf{G}^{(n)} - \theta_j))] - \boldsymbol{E}[g(\max_{j \in [k]}(V^j \cdot \mathbf{G}^{(d)} - \theta_j))] \right| \le O(\|g''\|_\infty \varepsilon' + \|g'\|_\infty \sqrt{\varepsilon' \ln k}).$$

We use the rest of this subsection to prove Theorem 17. The proof extends the proofs of similar results in [4, 6] to the case of non-centered Gaussians.

For ease of presentation, for $i \in [k]$ define the non-centered Gaussian random variables $\mathbf{X}_i := W^i \cdot \mathbf{G}^{(n)} - \theta_i$ and $\mathbf{Y}_i := V^i \cdot \mathbf{G}^{(d)} - \theta_i$. We may suppose, without loss of generality, that $\mathbf{X} = (\mathbf{X}_1, \dots, \mathbf{X}_k)$ and $\mathbf{Y} = (\mathbf{Y}_1, \dots, \mathbf{Y}_k)$ are defined over the same probability space and that $\mathbf{X}$ and $\mathbf{Y}$ are independent of each other. Our goal is to bound the magnitude of the difference

$$\mathbf{E}[g(F_\beta(\mathbf{X}_1, \dots, \mathbf{X}_k))] - \mathbf{E}[g(F_\beta(\mathbf{Y}_1, \dots, \mathbf{Y}_k))]. \tag{22}$$

Let μ_i denote $\mathbf{E}[\mathbf{X}_i] = \mathbf{E}[\mathbf{Y}_i]$, and let $\widetilde{\mathbf{X}}_i = \mathbf{X}_i - \mu_i$ be the centered version of $\mathbf{X}_i$ and similarly let $\widetilde{\mathbf{Y}_i} = \mathbf{Y}_i - \mu_i$. Observe that by independence we have $\mathbf{E}[\mathbf{X}_i \mathbf{Y}_j] = 0$ for all $i, j \in [k]$. Now, as is standard, we do a Slepian interpolation; so for $t \in [0, 1]$, we define

$\mathbf{Z}_{t,i} := \sqrt{t}\widetilde{\mathbf{X}}_i + \sqrt{1-t}\widetilde{\mathbf{Y}}_i + \mu_i$, and we write $\mathbf{Z}_t$ to denote $(\mathbf{Z}_{t,1}, \ldots, \mathbf{Z}_{t,k})$. We define the function

$$\Psi(t) = \mathbf{E}[g(F_\beta(\mathbf{Z}_{t,1}, \ldots, \mathbf{Z}_{t,k}))],$$

and we observe that

$$(22) = \Psi(1) - \Psi(0) = \int_0^1 \Psi'(t)dt. \tag{23}$$

Thus to upper bound the magnitude of (22) it suffices to upper bound $\int_0^1 |\Psi'(t)|dt$.

For $x \in \mathbb{R}^k$ let us write $m(x)$ to denote $g(F_\beta(x))$. By applying the chain rule, we have

$$\Psi'(t) = \frac{1}{2}\sum_{i=1}^{k} \mathbf{E}\left[\partial_i m(\mathbf{Z}_t) \cdot \left(\frac{\widetilde{\mathbf{X}_i}}{\sqrt{t}} - \frac{\widetilde{\mathbf{Y}_i}}{\sqrt{1-t}}\right)\right].$$

Now we recall the following "integration by parts" lemma, which is sometimes referred to as "Stein's identity:"

▶ **Lemma 18** (Lemma 2 of [6], see also Lemma 2.1 of [4]). *Let $\mathbf{A} = (\mathbf{A}_1, \ldots, \mathbf{A}_p)$ be a p-dimensional Gaussian random vector with mean zero and let $f : \mathbb{R}^p \to \mathbb{R}$ be a C^1 function with $\boldsymbol{E}[|\partial_\ell f(\mathbf{A})|] < \infty$ for all $\ell \in [p]$. Then for each $\ell \in [p]$, we have*

$$\boldsymbol{E}[\mathbf{A}_\ell f(\mathbf{A})] = \sum_{j=1}^{p} \boldsymbol{E}[\mathbf{A}_\ell \mathbf{A}_j]\, \boldsymbol{E}[\partial_j f(\mathbf{A})].$$

We now set (i) $p = k+1$, (ii) $\mathbf{A}_j = \mathbf{Z}_{t,j}$ (for $1 \le j \le k$), (iii) $\mathbf{A}_{k+1} = \frac{\tilde{\mathbf{X}}_i}{\sqrt{t}} - \frac{\tilde{\mathbf{Y}}_i}{\sqrt{1-t}}$ and (iv) $f(\mathbf{A}) = \partial_i m(\mathbf{Z}_t)$. Observe that with this setting, $\partial_{k+1} f(\mathbf{A}) = 0$. Applying Lemma 18 with $\ell = k+1$, we get that

$$\begin{aligned}\Psi'(t) &= \frac{1}{2}\sum_{i=1}^{k}\sum_{j=1}^{k} \mathbf{E}\left[\left(\frac{\tilde{\mathbf{X}}_i}{\sqrt{t}} - \frac{\tilde{\mathbf{Y}}_i}{\sqrt{1-t}}\right)\left(\sqrt{t}\tilde{\mathbf{X}}_j - \sqrt{1-t}\tilde{\mathbf{Y}}_j\right)\right] \mathbf{E}[\partial_{i,j} m(\mathbf{Z}_t)] \\ &= \frac{1}{2}\sum_{i=1}^{k}\sum_{j=1}^{k} (\sigma^W_{i,j} - \sigma^V_{i,j}) \cdot \mathbf{E}[\partial_{i,j} m(\mathbf{Z}_t)],\end{aligned}$$

where the second equality uses the independence between $\mathbf{X}$ and $\mathbf{Y}$. We get that

$$\begin{aligned}\int_{t=0}^{1} |\Psi'(t)|dt &\le \frac{1}{2}\int_{t=0}^{1} \sum_{i,j=1}^{k} |\sigma^W_{i,j} - \sigma^V_{i,j}| \cdot |\mathbf{E}[\partial_{i,j} m(\mathbf{Z}_t)]|\ dt \\ &\le \frac{\Delta}{2} \cdot \int_{t=0}^{1} \sum_{i,j=1}^{k} |\mathbf{E}[\partial_{i,j} m(\mathbf{Z}_t)]|\ dt,\end{aligned} \tag{24}$$

where $\Delta = \max_{i,j\in[k]} |\sigma^W_{i,j} - \sigma^V_{i,j}|$ is the quantity defined in (9). Thus, we are left with the task of upper bounding the double derivatives. We have

$$\partial_i m(x) = \partial_i(g(F_\beta(x_1, \ldots, x_k))) = g'(F_\beta(x_1, \ldots, x_k)) \cdot \frac{\partial F_\beta}{\partial x_i}$$

and hence

$$\partial_{i,j} m(x) = \partial_{i,j}(g(F_\beta(x_1, \ldots, x_k))) = g''(F_\beta(x_1, \ldots, x_k)) \cdot \frac{\partial F_\beta}{\partial x_i}\frac{\partial F_\beta}{\partial x_j} + g'(F_\beta(x_1, \ldots, x_k)) \cdot \frac{\partial^2 F_\beta}{\partial x_i \partial x_j}.$$

Applying Facts 15 and 16, it follows that

$$\sum_{i,j=1}^{k} |\mathbf{E}[\partial_{i,j} m(\mathbf{Z}_t)]| = O(\|g''\|_\infty + \|g'\|_\infty \cdot \beta).$$

Hence combining (23), (24), and the above, and recalling that $\Delta \leq 9\varepsilon'$ (see Observation 6), we get that

$$|\mathbf{E}[g(F_\beta(\mathbf{X}_1, \ldots, \mathbf{X}_k))] - \mathbf{E}[g(F_\beta(\mathbf{Y}_1, \ldots, \mathbf{Y}_k))]| \leq O(||g''||_\infty \cdot \varepsilon' + ||g'||_\infty \cdot \varepsilon' \cdot \beta),$$

giving the first claim of the theorem. For the second claim, using Fact 14, it follows that

$$\begin{aligned} |\,\mathbf{E}[g(\max_{j\in[k]}(\mathbf{X}_j))] - \mathbf{E}[g(\max_{j\in[k]}(\mathbf{X}_j))]| &\leq O(||g''||_\infty \cdot \varepsilon' + ||g'||_\infty \cdot \varepsilon' \cdot \beta) + ||g'||_\infty \cdot \frac{\ln k}{\beta} \\ &\leq O\Big(||g'||_\infty \cdot \big(\varepsilon' \cdot \beta + (\ln k)/\beta\big) + ||g''||_\infty \cdot \varepsilon'\Big). \end{aligned}$$

The second claim of the theorem now follows by setting $\beta = \sqrt{(\ln k)/\varepsilon'}$.

6.3 Formalizing step (3') of the intuitive sketch: anticoncentration of max of non-centered Gaussians

We recall the following useful anticoncentration result from [16], which follows almost directly from a result of Nazarov [27]:

▶ **Lemma 19** (Lemma 3.4 of [16]: anticoncentration of multidimensional Gaussian)**.** *Let $W^1, \ldots, W^k$ be unit vectors in $\mathbb{R}^n$. For all $\vec{\theta} \in \mathbb{R}^k$ and all $\lambda > 0$, we have*

$$\underset{\mathbf{G}\leftarrow\mathcal{N}(0,1)^n}{\boldsymbol{Pr}} \left[\, W\mathbf{G} \in \mathrm{Strip}_{\lambda,k,\vec{\theta}} \right] = O(\lambda\sqrt{\log k}). \tag{25}$$

This can be viewed as a k-dimensional analogue of Theorem 3 from [6], which gives an anticoncentration bound on $\max\{W^1 \cdot \mathbf{G}, \cdots W^k \cdot \mathbf{G}\}$ (and also the above lemma is for non-centered Gaussians, whereas Theorem 3 of [6] is about centered Gaussians). As an immediate consequence of Lemma 19 we obtain the following:

▶ **Theorem 20** (anticoncentration of max of non-centered Gaussians)**.** *Fix any $\vec{\theta} \in \mathbb{R}^k$. For all $\lambda > 0$ and all $t \in \mathbb{R}$ it holds that*

$$\boldsymbol{Pr}[\max_{j\in[k]} (W^j \cdot \mathbf{G}^{(n)} - \theta_j) \in [t - \lambda, t]] = O(\lambda\sqrt{\log k}).$$

6.4 Formalizing step (4') of the intuitive sketch: Passing from a smooth approximator of $\mathrm{sign}(\cdot)$ to $\mathrm{sign}(\cdot)$

In this section we prove the following theorem, which upper bounds the Kolmogorov distance between the random variables $\max_{j\in[k]} (W^j \cdot \mathbf{G}^{(n)} - \theta_j)$ and $\max_{j\in[k]} (V^j \cdot \mathbf{G}^{(d)} - \theta_j)$:

▶ **Theorem 21.** *Let $W^1, \ldots, W^k$ be unit vectors in $\mathbb{R}^n$, $V^1, \ldots, V^k$ be vectors in $\mathbb{R}^d$ satisfying (8). For all $\vec{\theta} \in \mathbb{R}^k$, the following bound holds:*

$$d_{\mathrm{K}}\left(\max_{j\in[k]} (W^j \cdot \mathbf{G}^{(n)} - \theta_j), \max_{j\in[k]} (V^j \cdot \mathbf{G}^{(d)} - \theta_j)\right) \leq O(\varepsilon' \log^2 k)^{1/3}.$$

This is equivalent to showing that for all $\vec{\theta} \in \mathbb{R}^k$ and all $t \in \mathbb{R}$, we have

$$|\mathbf{Pr}[\max_{j\in[k]} (W^j \cdot \mathbf{G}^{(n)} - \theta_j) \le t] - \mathbf{Pr}[\max_{j\in[k]} (V^j \cdot \mathbf{G}^{(d)} - \theta_j) \le t]| \le O(\varepsilon' \log^2 k)^{1/3}. \quad (26)$$

Our argument follows the proof of Theorem 2 in [6]; the main idea is to combine Theorem 17, where g is a smooth approximation of the sign function, with Theorem 20, which establishes anticoncentration of the max of non-centered Gaussians. The particular $g \in C^2(\mathbb{R}), g : \mathbb{R} \to [-1, 1]$ which we use is the following smooth approximator of the sign function:

$$g(z) = \begin{cases} -1 & z \le -1 \\ -60 \int_{(z+1)/2}^{1} s^2(1-s)^2 ds + 1 & -1 < z < 1 \\ 1 & z \ge 1. \end{cases}$$

Given parameters $x \in \mathbb{R}, \beta > 0$, and $\delta > 0$, define the function $g_{x,\beta,\delta}(z) = g((z - x - e_\beta)/\delta)$. We record a simple claim that can be verified by direct calculation:

▷ Claim 22. For any $x \in \mathbb{R}$, $\beta > 0$ and $\delta > 0$, the following hold:

1. $||g'_{x,\beta,\delta}||_\infty = ||g'||_\infty/\delta \le O(1/\delta)$,
2. $||g''_{x,\beta,\delta}||_\infty = ||g'||_\infty/\delta^2 \le O(1/\delta^2)$,
3. $\mathbb{1}(z \le x + e_\beta) \le g_{x,\beta,\delta}(z) \le \mathbb{1}(z \le x + e_\beta + \delta)$, for all $z \in \mathbb{R}$.

We now proceed to prove (26). As before, for ease of presentation define the random variables $\mathbf{X}_i = W^i \cdot \mathbf{G}^{(n)} - \theta_i$ and $\mathbf{Y}_i = V^i \cdot \mathbf{G}^{(d)} - \theta_i$, $i \in [k]$.

For arbitrary $x \in \mathbb{R}, \beta > 0$, and $\delta > 0$, we have

$$\begin{aligned}
\mathbf{Pr}\left[\max_{j\in[k]} \mathbf{X}_j \le x\right] &\le \mathbf{Pr}[F_\beta(\mathbf{X}) \le x + e_\beta] && \text{(Claim 14)} \\
&\le \mathbf{E}[g_{x,\beta,\delta}(F_\beta(\mathbf{X}))] && \text{(Claim 22)} \\
&\le \mathbf{E}[g_{x,\beta,\delta}(F_\beta(\mathbf{Y}))] + O\left(||g''||_\infty \cdot \frac{\varepsilon'}{\delta^2} + ||g'||_\infty \cdot \frac{\varepsilon'\beta}{\delta}\right) && \text{(Theorem 17, Claim 22)} \\
&\le \mathbf{Pr}[F_\beta(\mathbf{Y}) \le x + e_\beta + \delta] + O\left(\frac{\varepsilon'}{\delta^2} + \frac{\varepsilon'\beta}{\delta}\right) && \text{(Claim 22)} \\
&\le \mathbf{Pr}[\max_{j\in[k]} \mathbf{Y}_j \le x + e_\beta + \delta] + e_\beta + O\left(\frac{\varepsilon'}{\delta^2} + \frac{\varepsilon'\beta}{\delta}\right) && \text{(Claim 14)} \\
&= \mathbf{Pr}[\max_{j\in[k]} \mathbf{Y}_j \le x] + (\mathbf{Pr}[\max_{j\in[k]} \mathbf{Y}_j \le x + e_\beta + \delta] - \mathbf{Pr}[\max_{j\in[k]} \mathbf{Y}_j \le x]) + \\
&\quad e_\beta + O\left(\frac{\varepsilon'}{\delta^2} + \frac{\varepsilon'\beta}{\delta}\right) \\
&\le \mathbf{Pr}[\max_{j\in[k]} \mathbf{Y}_j \le x] + O((e_\beta + \delta)\sqrt{\log k}) + e_\beta + O\left(\frac{\varepsilon'}{\delta^2} + \frac{\varepsilon'\beta}{\delta}\right) && \text{(Theorem 20)}
\end{aligned}$$

Setting $\beta = (\log k)/\delta$ and $\delta = O(\varepsilon'\sqrt{\log k})^{1/3}$ completes the proof of (26).

6.5 Formalizing step (5') of the intuitive sketch: Re-interpreting the Kolmogorov distance bound as a PRG

We conclude the proof of our PRG construction from the bound proved in Theorem 21; recall that this gives CDF-closeness at every point in $\mathbb{R}$, specifically

$$d_{\mathrm{K}}(\max_{j\in[k]} (W^j \cdot \mathbf{G}^{(n)} - \theta_j), \max_{j\in[k]} (V^j \cdot \mathbf{G}^{(d)} - \theta_j)) \le O(\varepsilon' \log^2 k)^{1/3}$$

Specializing this to CDF-closeness at the point 0, we get that

$$\left|\mathbf{Pr}[W^j \cdot \mathbf{G}^{(n)} \le \theta_j \text{ for all } j \in [m]] - \mathbf{Pr}[V^j \cdot \mathbf{G}^{(d)} \le \theta_j \text{ for all } j \in [m]]\right| \le O(\varepsilon' \log^2 k)^{1/3}.$$

Now we recall that, from Section 4.2, all but a $\delta' = \varepsilon$ fraction of outcomes $V^1, \ldots, V^k$ of $\mathbf{V}^j = W^j \mathbf{A}^\mathsf{T}$ satisfy (8). Hence we have

$$\left|\mathbf{Pr}[W^j \cdot \mathbf{G}^{(n)} \leq \theta_j \text{ for all } j \in [m]] - \mathbf{Pr}[W^j \mathbf{A}^\mathsf{T} \cdot \mathbf{G}^{(d)} \leq \theta_j \text{ for all } j \in [m]]\right| \leq O(\varepsilon' \log^2 k)^{1/3} + \varepsilon,$$

and recalling that a draw $\mathbf{Z}$ from our generator Gen is $\mathbf{Z} = \mathbf{A}^\mathsf{T}\mathbf{G}^{(d)}$, we get that this is equivalent to

$$\left|\mathbf{Pr}[W^j \cdot \mathbf{G}^{(n)} \leq \theta_j \text{ for all } j \in [m]] - \mathbf{Pr}[W^j \cdot \mathbf{Z} \leq \theta_j \text{ for all } j \in [m]]\right| \leq O(\varepsilon' \log^2 k)^{1/3} + \varepsilon.$$

Setting $\varepsilon' = \varepsilon^3 / \log^2 k$ completes the proof of correctness of our PRG construction.

7 Application of our PRG: Deterministic approximate counting for functions of LTFs over $\{-1, 1\}^n$

In this section we prove Theorems 4 and 5, which we state with precise bounds as two parts of the following theorem.

▶ **Theorem 23** (Restatements of Theorem 4 and 5)**.**

1. (Arbitrary functions of LTFs). *There is a deterministic algorithm which, given as input k LTFs $h_1, \ldots, h_k$ over $\{-1,1\}^n$, an explicit function $g : \{-1,1\}^k \to \{-1,1\}$, and an error parameter $\varepsilon > 0$, runs in $\mathrm{poly}(n) \cdot 2^{\tilde{O}(\frac{k^{15}}{\varepsilon^{12}})}$ time and outputs a value $\tilde{v} \in [0,1]$ such that $|\tilde{v} - v| \leq \varepsilon$, where v is the fraction of points in $\{-1,1\}^n$ that satisfy $g(h_1, \ldots, h_k)$.*
2. (Intersections of LTFs). *There is a deterministic algorithm which, given as input k LTFs $h_1, \ldots, h_k$ over $\{-1,1\}^n$ and an error parameter $\varepsilon > 0$, runs in $\mathrm{poly}(n) \cdot 2^{\mathrm{poly}(\log k, 1/\varepsilon)}$ time and outputs a value $\tilde{v} \in [0,1]$ such that $|\tilde{v} - v| \leq \varepsilon$, where v is the fraction of points in $\{-1,1\}^n$ that satisfy $h_1(x) \wedge \cdots \wedge h_k(x)$.*

We prove Part 1 first since it is simpler and relies on (extensions of) known tools such as regularity lemmas and invariance principles. In particular, Part 1 requires an invariance principle for arbitrary functions of LTFs. Such an invariance principle was proved in [14]; we provide an alternate proof of the invariance principle that we require in Appendix C, which we believe could be of independent interest. For Part 2, the main ingredients are an invariance principle of [16] for intersections of LTFs and a "multi-regularity lemma" for k-tuples of LTFs due to [14] along with a subtle application of the PRG for intersections of LTFs due to [28].

7.1 A useful notion: Regularity

Given an LTF $h(x) = \mathrm{sign}(w_1 x_1 + \cdots + w_n x_n - \theta)$ and a parameter $0 < \tau < 1$, we say that h is *τ-regular* if

$$\sum_{j=1}^n w_j^4 \leq \tau^2 \cdot (\sum_{j=1}^n w_j^2)^2.$$

Intuitively, τ-regularity (when τ is small) captures the property that no weight in $w_1, \ldots, w_n$ has magnitude which is large relative to "the overall scale of the weights." Regularity is a useful condition because if w is a τ-regular weight vector with two-norm 1, then by the Berry-Esseen theorem [1, 10] the CDF of the real random variable $w \cdot \mathbf{X}$ (where $\mathbf{X}$ is uniform over $\{-1,1\}^n$) is τ-close to the CDF of an $\mathcal{N}(0,1)$ Gaussian. Thus the Berry-Esseen theorem implies that regular LTFs will "behave similarly" whether they are given uniform inputs $\mathbf{X} \leftarrow \{-1,1\}^n$ or Gaussian inputs $\mathbf{G} \leftarrow \mathcal{N}(0,1)^n$; in this sense, it can be viewed as an invariance principle for a single LTF.

7.2 Proof of Part 1 of Theorem 23: Arbitrary functions of k LTFs

The first principal ingredient that we use is an *invariance principle for arbitrary functions of LTFs*. As mentioned earlier, such a result was established in [14] via a "Lindeberg-method" type proof. In Appendix C we give an alternate proof (which is very different from the proofs of [14, 16]) of the version that we require, which is stated below:

▶ **Theorem 24** (Invariance principle for arbitrary functions of k LTFs). *Let $h_1, h_2, \ldots, h_k$ be τ-regular LTFs and let $F(x) = g(h_1(x), \cdots, h_k(x))$ where $g : \{-1,1\}^k \to \{-1,1\}$ may be any function. Then*

$$\left| \mathop{\boldsymbol{Pr}}_{\mathbf{X} \leftarrow \{-1,1\}^n} [F(\mathbf{X}) = -1] - \mathop{\boldsymbol{Pr}}_{\mathbf{Z} \leftarrow \mathcal{N}(0,1)^n} [F(\mathbf{Z}) = -1] \right| \leq O(k^{3/2} \tau \sqrt{\log(k/\tau)}). \tag{27}$$

Combining Theorem 2 (our PRG for arbitrary functions of LTFs over Gaussian space) and Theorem 24, an algorithm that simply enumerates over all the seeds of our PRG yields the following deterministic approximate counting algorithm for intersections of sufficiently regular LTFs:

▶ **Corollary 25** (Deterministic approximate counting for arbitrary functions of regular LTFs). *There is a deterministic algorithm with the following performance guarantee: Given $\varepsilon > 0$, a collection $h_1, \ldots, h_k$ of LTFs over $\{-1,1\}^n$ each of which is τ-regular where $\tau = O(\frac{\varepsilon}{k^{3/2}\sqrt{(\log k)(\log \frac{k}{\varepsilon})}})$, and a function $g : \{-1,1\}^k \to \{-1,1\}$, the algorithm runs in time $\mathrm{poly}(n) \cdot 2^{\tilde{O}(\frac{k^{15}}{\varepsilon^{12}})}$ and outputs a value $\tilde{v} \in [0,1]$ such that $|\tilde{v} - v| \leq \varepsilon$, where v is the fraction of points in $\{-1,1\}^n$ that satisfy $g(h_1, \ldots, h_k)$.*

We next extend Corollary 25 to obtain a deterministic approximate counting algorithm for arbitrary functions of k *general* LTFs using a slight extension of the "multi-regularity lemma" established in [14] (see Theorem 5.4 of the ArXiV version, available at [15]) for k-tuples of general LTFs.

While not precisely stated in these terms, we recall that this multi-regularity lemma, roughly speaking, asserts the following: Given a k-tuple of LTFs $h_1, \ldots, h_k$, there is a relatively shallow non-adaptive decision tree on the variables such that for all $i \in [k]$, one of the two following two possibilities hold:

1. For every leaf ρ of the decision tree (corresponding to a restriction), the restricted LTF $h_i \restriction \rho$ is regular.
2. With high probability, the restricted LTF $h_i \restriction \rho$ is close to a constant.

Similar to Lemma 18 of [7], the multi-regularity lemma of [14] can be implemented as a deterministic algorithm. In fact, because the decision tree is *non-adaptive*, the set of variables appearing in the internal nodes can be computed in time polynomial in depth of the decision tree (as opposed to exponential in the depth which is the size of the tree). This is because at each node, in order to choose which variable from $x_1, \ldots, x_n$ should be placed at that node it suffices to compute the influence of each variable in each of the k restricted linear forms, and this is a straightforward deterministic computation. We remark that the ability to *compute the tree* in polynomial time (in terms of its depth) is not crucial for this subsection. However, it is vital for the application in the next subsection – deterministic counting for intersections of general LTFs. Viewed as an algorithmic procedure from this perspective, Theorem 5.4 of [15] yields the following in our setting:

▶ **Lemma 26** (Algorithmic regularity lemma for LTFs, general k, based on Theorem 5.4 of [15]). *There is an algorithm* `ConstructTree` *with the following properties: Let $h_1, \ldots, h_k$ be LTFs over $\{-1,1\}^n$. Algorithm* `ConstructTree` *(which is deterministic) receives $h_1, \ldots, h_k$ and*

$0 < \tau, \gamma < 1/4$ as input, runs in time $\mathrm{poly}(n, D_k(\tau,\gamma))$ *and outputs a set of variables* $S \subseteq [n]$ *and a* k*-tuple of labels* $(\mathrm{label}_1, \ldots, \mathrm{label}_k) \in \{R, J\}^k$ *such that the following holds:*

1. $|S| \leq D_k(\tau,\gamma)$ *where*

$$D_k(\tau,\gamma) := k \cdot \frac{1}{\tau} \cdot \mathrm{poly}\left(\log \frac{1}{\gamma}\right).$$

2. *For each leaf* ρ *and* $i \in [k]$*, if* $\mathrm{label}_i = R$*, then the LTF* $h_i \restriction \rho$ *is* τ*-regular.*
3. *For each* $i \in [k]$*, if* $\mathrm{label}_i = J$*, then the LTF* h'_i *obtained by zeroing the coordinates outside* S *satisfies* $\boldsymbol{Pr}_{x\in\{-1,1\}^n}[h_i(x) \neq h'_i(x)] \leq \gamma$*. In particular, observe that for any leaf* ρ*,* $h'_i \restriction \rho$ *is fixed at either* $+1$ *or* -1*.*

▶ Remark 27. The theorem above can be obtained by essentially observing the proof of Theorem 5.4 in [15]. In particular, the S in the above theorem corresponds to the H_0 in their theorem. Similarly, the coordinates $i \in [k]$ which are labeled 'R' (resp. labeled 'J') in our theorem correspond exactly to the coordinates $i \in [d]$ which fall in the first case (resp. second case) of Theorem 5.4 in [15]. To get the guarantee for the third case, we define h'_i as follows. Let $h_i(x) = \mathrm{sign}(\sum_j w_{i,j}x_j - \theta_j)$. We then define $h'_i(x) = \mathrm{sign}(\sum_{j\in S} w_{i,j}x_j - \theta_j)$, i.e., simply erase the coordinates outside of S. The upper bound on the quantity $\mathbf{Pr}_{x\in\{-1,1\}^n}[h_i(x) \neq h'_i(x)]$ can essentially be derived from the event whose probability is upper bound in the centered equation in item (2) of Theorem 5.4 of [15].

We now extend the algorithm in Corollary 29 to handle arbitrary functions of k general LTFs using the algorithmic regularity lemma for multiple LTFs given in Lemma 26. The parameter"δ" in Lemma 26 is set to ε and the parameter "γ" is set to ε/k, and the parameter "τ" is set to $O(\frac{\varepsilon}{k^{3/2}\sqrt{(\log k)(\log \frac{k}{\varepsilon})}})$ so that Corollary 25 can be applied. Constructing the decision tree in the first step of the algorithm for general LTFs takes time $\mathrm{poly}(n, D_k(\tau,\varepsilon,\delta)) = \mathrm{poly}(n, k, 1/\varepsilon)$. In the second step of the algorithm for general LTFs, for each leaf ρ in the decision tree,

- If any of the k labels are "fail" the contribution from that leaf is 0;
- If all k labels are bits $b_1, \ldots, b_k \in \{-1, 1\}$, then the contribution from that leaf is $2^{-D_k} \cdot \mathbb{1}[g(b_1, \ldots, b_k) = -1]$;
- If $k - t$ of the labels (for notational convenience, say these are the ones corresponding to $h_{t+1}, \ldots, h_k$) are bits $b_{t+1}, \ldots, b_k$ and the remaining t labels (say the ones corresponding to $h_1 \restriction \rho, \ldots, h_t \restriction \rho$) are "regular," we run the approximate counting algorithm for the regular case from Corollary 25 to compute an $\pm\varepsilon$-accurate estimate (call it v_ρ) of the fraction of satisfying assignments of $g((h_1 \restriction \rho) \wedge \cdots \wedge (h_t \restriction \rho), b_{t+1}, \ldots, b_k)$, and the contribution from that leaf is $2^{-D_k} \cdot v_\rho$.

The overall running time for the algorithm is at most $\mathrm{poly}(n)\cdot$ (number of leaves) $\cdot$ (running time of Corollary 25), which is $\mathrm{poly}(n) \cdot 2^{\tilde{O}(k^{3/2}/\varepsilon)+\tilde{O}(k^{15}/\varepsilon^{12})}$. To establish correctness, we observe that the final value $\tilde{v}$ may be viewed as a sum of contributions across all the leaves. Property 3 of Lemma 26 and the setting of $\delta = \varepsilon$ in Step 1 ensures that leaves that have any "fail" label contribute a total of $O(\varepsilon)$ to the error $|v - \tilde{v}|$. The setting of the γ parameter to be ε/k ensures that leaves containing any $+1$ label, or having all -1's as their labels, collectively contribute a total of at most $O(\varepsilon)$ to $|v - \tilde{v}|$. Finally, Theorem 2 ensures that leaves as in the last bullet above contribute a total of $O(\varepsilon)$ to $|v - \tilde{v}|$. This concludes the proof of Theorem 23.

7.3 Proof of Part 2 of Theorem 23: Intersections of k LTFs

We begin by recalling the main structural result of [16], which extends the Berry-Esseen theorem to *intersections* of LTFs (also known as polytopes). (Recall that we view -1 as "true" and $+1$ as "false.")

▶ **Theorem 28** (Theorem 3.1 of [16]: invariance principle for polytopes). *Let $h_1, h_2, \ldots, h_k$ be τ-regular LTFs and let $F(x) = h_1(x) \wedge \cdots \wedge h_k(x)$. Then*

$$\left| \mathop{\boldsymbol{Pr}}_{\mathbf{U}^{(n)} \leftarrow \{-1,1\}^n} [F(\mathbf{U}^{(n)}) = -1] - \mathop{\boldsymbol{Pr}}_{\mathbf{G}^{(n)} \leftarrow \mathcal{N}(0,1)^n} [F(\mathbf{G}^{(n)}) = -1] \right| \leq C(\log k)^{8/5}(\tau \log(1/\tau))^{1/5}$$

where C is an absolute constant.

Combining Theorem 3 (our PRG for intersections of LTFs over Gaussian space) and Theorem 28, an algorithm that simply enumerates over all the seeds of our PRG yields the following deterministic approximate counting algorithm for intersections of sufficiently regular LTFs:

▶ **Corollary 29** (Deterministic approximate counting for intersections of regular LTFs). *There is a deterministic algorithm with the following performance guarantee: Given $\varepsilon > 0$ and a collection $h_1, \ldots, h_k$ of LTFs over $\{-1,1\}^n$, each of which is τ-regular where $\tau = O(\frac{\varepsilon^5}{\log^8(k) \cdot \log(\frac{\log k}{\varepsilon})})$, the algorithm runs in time $\mathrm{poly}(n) \cdot 2^{\tilde{O}(\frac{\log^6 k}{\varepsilon^6})}$ and outputs a value $\tilde{v} \in [0,1]$ such that $|\tilde{v} - v| \leq \varepsilon$, where v is the fraction of points in $\{-1,1\}^n$ that satisfy $h_1 \wedge \cdots \wedge h_k$.*

The above algorithm works only for intersections of sufficiently regular LTFs. We will now extend Corollary 29 to obtain a deterministic approximate counting algorithm for intersections of k *general* LTFs using two tools. The first is the multi-regularity lemma (Lemma 26) from the previous subsection. The second ingredient we require is the recent construction of a PRG for intersection of LTFs by O'Donnell, Servedio and Tan [28] where they construct a PRG for the uniform distribution on $\{-1,1\}^n$ which fools intersections of k LTFs with seed length $(\log n) \cdot \mathrm{poly}(\log k, 1/\varepsilon)$. More precisely, we have the following theorem from [28].

▶ **Theorem 30.** *There is an efficiently computable ε-PRG $\mathcal{G}_{\mathsf{OST}} : \{-1,1\}^s \to \{-1,1\}^n$ for intersections of k LTFs over $\{-1,1\}^n$ with $s = (\log n) \cdot \mathrm{poly}(\log k, 1/\varepsilon)$.*

Observe that while $\mathcal{G}_{\mathsf{OST}}$ simultaneously achieves polylogarithmic dependence on both n and k, to get a deterministic approximate counting algorithm with the kind of guarantee we want, we would need a seed length of the form $\log n + \mathrm{poly}(\log k, 1/\varepsilon)$. While we will crucially use $\mathcal{G}_{\mathsf{OST}}$, we will essentially bootstrap it with the algorithms from Corollary 29 and Lemma 26 as follows.

Given as input $h_1, \ldots, h_k$ and a desired accuracy parameter ε, the algorithm proceeds as follows:

1. Run the algorithm `ConstructTree` on the LTFs $h_1, \ldots, h_k$ with its "γ" parameter as $\varepsilon/4k$ and "τ" parameter as $O(\frac{\varepsilon^5}{\log^8(k) \cdot \log(\frac{\log k}{\varepsilon})})$.
2. Let S be the set of variables returned by the algorithm `ConstructTree`. We set $n_{\mathsf{OST}} = |S|$, $\varepsilon_{\mathsf{OST}} = \varepsilon/4$ and $k_{\mathsf{OST}} = k$.
3. Let us run $\mathcal{G}_{\mathsf{OST}}$ with parameters n_{OST}, $\varepsilon_{\mathsf{OST}}$ and k_{OST}. Let s_{OST} be the seed length. Let $\mathcal{O}_{\mathsf{OST}} \subseteq \{-1,1\}^S$ denote the range of $\mathcal{G}_{\mathsf{OST}}$. We treat each $\rho \in \mathcal{O}_{\mathsf{OST}}$ as an assignment for the coordinates in S.

4. For each $\rho \in \mathcal{O}_{\mathsf{OST}}$, compute v_ρ as follows: If there is any $i \in [k]$ such that $\text{label}_i = J$ and $h'_i \restriction \rho = -1$, then set $v_\rho = 0$. Otherwise, observe that for all $i \in [k]$ such that $\text{label}_i = R$, $h_i \restriction \rho$ is τ-regular (for τ specified earlier). Run the algorithm from Corollary 29 to compute $\mathbf{Pr}[\wedge_{i:\text{label}_i=R}(h_i \restriction \rho)]$. Let the output be v_ρ.
5. Output the value $\mathbf{E}_{\rho \in \mathcal{O}_{\mathsf{OST}}}[v_\rho]$.

The analysis of the running time of the above routine is straightforward: Observe that for our choice of τ and γ, the value $D_k(\tau, \gamma)$ (from Lemma 26) is $\tilde{O}(k \cdot \varepsilon^{-5})$. The running time of the first step, i.e., `ConstructTree` is bounded by $\text{poly}(n, D_k(\tau, \gamma))$. Now, observing that $|S| \leq D_k(\tau, \gamma)$, from Theorem 30, we get that $s_{\mathsf{OST}} = \mathsf{poly}(\log k, \varepsilon^{-1})$ and thus $|\mathcal{O}_{\mathsf{OST}}| = 2^{\mathsf{poly}(\log k, \varepsilon^{-1})}$. For each $\rho \in \mathcal{O}_{\mathsf{OST}}$, the running time of the algorithm from Lemma 26 is bounded by $\text{poly}(n) \cdot 2^{\tilde{O}(\log^6 k/\varepsilon^6)}$. Thus, the total running time is $|\mathcal{O}_{\mathsf{OST}}| \cdot \text{poly}(n) \cdot 2^{\tilde{O}(\log^6 k/\varepsilon^6)}$ which is $\text{poly}(n) \cdot 2^{\text{poly}(\log k, 1/\varepsilon)}$.

We now move to the proof of correctness of the algorithm. Observe that if $\text{label}_i = J$ for any $i \in [k]$, then by guarantee of Lemma 26, $\mathbf{Pr}_{x \in \{-1,1\}^n}[h_i(x) \neq h'_i(x)] \leq \gamma$. Thus, if we define $\mathcal{A}_J = \{i \in [k] : \text{label}_i = J\}$ and $\mathcal{A}_R = \{i \in [k] : \text{label}_i = R\}$,

$$\left| \mathbf{Pr}_{x \in \{-1,1\}^n}[h_1(x) \wedge \ldots \wedge h_k(x)] - \mathbf{Pr}_{x \in \{-1,1\}^n}[\wedge_{i \in \mathcal{A}_J} h'_i(x) \ \wedge_{i \in \mathcal{A}_R} h_i(x)] \right| \leq k\gamma = \frac{\varepsilon}{4}. \quad (28)$$

Now, consider any assignment $z \in \{-1,1\}^{[n] \setminus S}$ of the variables in $[n] \setminus S$. Then, using the guarantee of $\mathcal{G}_{\mathsf{OST}}$, we get

$$\left| \mathbf{Pr}_{x \in \{-1,1\}^S}[\wedge_{i \in \mathcal{A}_J} h'_i \restriction z(x) \ \wedge_{i \in \mathcal{A}_R} h_i \restriction z(x)] - \mathbf{Pr}_{\rho \in \mathcal{O}_{\mathsf{OST}}}[\wedge_{i \in \mathcal{A}_J} h'_i \restriction z(\rho) \ \wedge_{i \in \mathcal{A}_R} h_i \restriction z(\rho)] \right| \leq \frac{\varepsilon}{4}.$$

Averaging over all possible values of $z \in \{-1,1\}^{[n] \setminus S}$ and combining with (28), we get

$$\left| \mathbf{Pr}_{x \in \{-1,1\}n}[h_1(x) \wedge \ldots \wedge h_k(x)] - \mathbf{Pr}_{z \in \{-1,1\}^{[n] \setminus S}, \rho \in \mathcal{O}_{\mathsf{OST}}}[\wedge_{i \in \mathcal{A}_J} h'_i(z, \rho) \wedge_{i \in \mathcal{A}_R} h_i(z, \rho)] \right| \leq \frac{\varepsilon}{2}. \quad (29)$$

Now, observe that for any $\rho \in \mathcal{O}_{\mathsf{OST}}$, $h'_i(z, \rho) = h'_i(\rho)$ (since h'_i does not depend on the variables outside S). Further, for each $i \in \mathcal{A}_R$, the LTF $h_i \restriction \rho$ is τ-regular. Consequently, for each choice of ρ, Step 4 of our routine outputs v_ρ such that

$$\left| \mathbf{Pr}_{z \in \{-1,1\}^{[n] \setminus S}}[\wedge_{i \in \mathcal{A}_J} h'_i(z, \rho) \ \wedge_{i \in \mathcal{A}_R} h_i(z, \rho)] - v_\rho \right| \leq \frac{\varepsilon}{2}.$$

Averaging it over all choices of ρ, we get that output in the final step $\mathbf{E}_{\rho \in \mathcal{O}_{\mathsf{OST}}}[v_\rho]$ satisfies

$$\left| \mathbf{E}_{\rho \in \mathcal{O}_{\mathsf{OST}}}[v_\rho] - \mathbf{Pr}_{z \in \{-1,1\}^{[n] \setminus S}, \rho \in \mathcal{O}_{\mathsf{OST}}}[\wedge_{i \in \mathcal{A}_J} h'_i(z, \rho) \ \wedge_{i \in \mathcal{A}_R} h_i(z, \rho)] \right| \leq \frac{\varepsilon}{2}.$$

Combining this with (29) finishes the proof.

References

1 Andrew C. Berry. The accuracy of the Gaussian approximation to the sum of independent variates. *Transactions of the American Mathematical Society*, 49(1):122–136, 1941.
2 R. Bhatia. *Matrix analysis*, volume 169. Springer, 2013.
3 Thomas Bonis. Rates in the Central Limit Theorem and diffusion approximation via Stein's Method. *arXiv preprint*, 2015. `arXiv:1506.06966`.
4 Sourav Chatterjee. An error bound in the Sudakov-Fernique inequality. *arXiv preprint*, 2005. `arXiv:math/0510424`.

5 Xi Chen, Rocco Servedio, and Li-Yang Tan. New algorithms and lower bounds for testing monotonicity. In *Proceedings of the 55th Annual Symposium on Foundations of Computer Science (FOCS 2014)*, pages 286–295, 2014.

6 Victor Chernozhukov, Denis Chetverikov, and Kengo Kato. Comparison and anti-concentration bounds for maxima of Gaussian random vectors. *Probability Theory and Related Fields*, 162(1-2):47–70, 2015.

7 Anindya De, Ilias Diakonikolas, and Rocco A. Servedio. Deterministic approximate counting for juntas of degree-2 polynomial threshold functions. In *Proceedings of the 29th Annual Conference on Computational Complexity (CCC)*, pages 229–240. IEEE, 2014.

8 Ilias Diakonikolas, Parikshit Gopalan, Rajesh Jaiswal, Rocco A. Servedio, and Emanuele Viola. Bounded Independence Fools Halfspaces. *SIAM Journal on Computing*, 39(8):3441–3462, 2010.

9 Ilias Diakonikolas, Daniel M. Kane, and Jelani Nelson. Bounded independence fools degree-2 threshold functions. In *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*, pages 11–20. IEEE, 2010.

10 Carl-Gustav Esseen. On the Liapunoff limit of error in the theory of probability. *Arkiv för matematik, astronomi och fysik*, A:1–19, 1942.

11 X. Fernique. Regularité des trajectoires des fonctions aléatoires gaussiennes. In *Ecole d'Eté de Probabilités de Saint-Flour IV—1974*, pages 1–96. Springer, 1975.

12 C. Givens and R. Shortt. A class of Wasserstein metrics for probability distributions. *The Michigan Mathematical Journal*, 31(2):231–240, 1984.

13 Parikshit Gopalan, Daniel Kane, and Raghu Meka. Pseudorandomness via the discrete fourier transform. In *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on*, pages 903–922. IEEE, 2015.

14 Parikshit Gopalan, Ryan O'Donnell, Yi Wu, and David Zuckerman. Fooling functions of halfspaces under product distributions. In *Computational Complexity (CCC), 2010 IEEE 25th Annual Conference on*, pages 223–234. IEEE, 2010.

15 Parikshit Gopalan, Ryan O'Donnell, Yi Wu, and David Zuckerman. Fooling Functions of Halfspaces under Product Distributions, 2010. `arXiv:1001.1593`.

16 Prahladh Harsha, Adam Klivans, and Raghu Meka. An invariance principle for polytopes. *Journal of the ACM (JACM)*, 59(6):29, 2012.

17 Daniel Kane. A Small PRG for Polynomial Threshold Functions of Gaussians. In *FOCS*, pages 257–266, 2011.

18 Daniel Kane. k-independent Gaussians fool polynomial threshold functions. In *IEEE Conference on Computational Complexity*, pages 252–261, 2011.

19 Daniel Kane. A pseudorandom generator for polynomial threshold functions of Gaussian with subpolynomial seed length. In *Proceedings of the 29th Annual Conference on Computational Complexity (CCC)*, pages 217–228, 2014.

20 Daniel Kane, Raghu Meka, and Jelani Nelson. Almost optimal explicit Johnson-Lindenstrauss families. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 628–639. Springer, 2011.

21 Daniel M. Kane. A Polylogarithmic PRG for Degree 2 Threshold Functions in the Gaussian Setting. In *30th Conference on Computational Complexity, CCC 2015*, pages 567–581, 2015.

22 Daniel M. Kane. A polylogarithmic PRG for degree 2 threshold functions in the Gaussian setting. In *LIPIcs-Leibniz International Proceedings in Informatics*, volume 33. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2015.

23 Pravesh K. Kothari and Raghu Meka. Almost Optimal Pseudorandom Generators for Spherical Caps. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*, pages 247–256, 2015.

24 H. Landau and L. Shepp. On the supremum of a Gaussian process. *Sankhyā: The Indian Journal of Statistics, Series A*, pages 369–378, 1970.

25 R. Meka. A polynomial time approximation scheme for computing the supremum of Gaussian processes. *Ann. Appl. Probab.*, 25(2):465–476, April 2015.

26 Raghu Meka and David Zuckerman. Pseudorandom generators for polynomial threshold functions. *SIAM Journal on Computing*, 42(3):1275–1301, 2013.

27 Fedor Nazarov. On the Maximal Perimeter of a Convex Set in $\mathbb{R}^n$ with Respect to a Gaussian Measure. In *Geometric Aspects of Functional Analysis: Israel Seminar 2001-2002*, pages 169–187, 2003.

28 R. O'Donnell, R.A. Servedio, and L.-Y. Tan. Fooling Polytopes. Available at `arXiv:1808.04035`, to appear in STOC 2019, 2018.

29 Rocco A. Servedio and Li-Yang Tan. Deterministic Search for CNF Satisfying Assignments in Almost Polynomial Time. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 813–823. IEEE, 2017.

30 V. Sudakov. *Geometric problems in the theory of infinite-dimensional probability distributions*, volume 141. American Mathematical Soc., 1979.

31 M. Talagrand. Majorizing measures: the generic chaining. *The Annals of Probability*, pages 1049–1103, 1996.

32 Gregory Valiant and Paul Valiant. Estimating the unseen: an $n/\log(n)$-sample estimator for entropy and support size, shown optimal via new CLTs. In *Proceedings of the 43rd Symposium on Theory of Computing (STOC)*, pages 685–694, 2011.

33 Alex Zhai. A high-dimensional CLT in W2 distance with near optimal convergence rate. *Probability Theory and Related Fields*, 170(3):821–845, April 2018.

A Lower bound on seed length for PRG fooling arbitrary functions of k LTFs

The following simple claim gives an $\Omega(\log n)$ lower bound even for $k = 1$:

▷ Claim 31. Let $\mathcal{G}$ be a 0.49-PRG for the class of all LTFs over Gaussian space $\mathcal{N}(0,1)^n$. Then the seed length of $\mathcal{G}$ is at least $\lfloor \log n \rfloor$.

Proof. Suppose that $\mathcal{G}$ is a generator with seed length $s \leq \lfloor \log n \rfloor - 1$. Let $S = \{v^1, \ldots, v^m\} \subset \mathbb{R}^n, |S| \leq n/2$ be the set of all points $\mathcal{G}(\{-1,1\}^s)$. Since $m < n$ there is a unit vector $w \in \mathbb{R}^n$ which hs orthogonal to all of $v^1, \ldots, v^m$; fix such a w. Fix any value $\kappa = o_n(1)$. It is easy to see that the LTF $f(x) = \text{sign}(w \cdot x - \kappa)$ has $\mathbf{Pr}_{\mathbf{G}^{(n)} \leftarrow \mathcal{N}(0,1)^n}[f(\mathbf{G}^{(n)}) = 1] = \frac{1}{2} - o_n(1)$, but each of $v^1, \ldots, v^m$ has $\text{sign}(w \cdot x - \kappa) = \text{sign}(-\kappa) = -1$, so $\mathbf{Pr}[f(\mathcal{G}(\mathbf{U}^{(s)})) = 1] = 0$. Hence $\mathcal{G}$ cannot be a 0.49-PRG for the class of all LTFs over Gaussian space. ◁

▷ Claim 32. Let $k \leq n$ and let $\mathcal{G}$ be a 0.49-PRG for the class of all functions $g(h_1, \ldots, h_k) : \mathbb{R}^n \to \{-1,1\}^n$ where $g : \{-1,1\}^k \to \{-1,1\}$ and each h_i is an LTF. Then the seed length of $\mathcal{G}$ is at least k.

Proof. Suppose that $\mathcal{G}$ is a generator with seed length $s \leq k - 1$. Let $S = \{v^1, \ldots, v^m\} \subset \mathbb{R}^n, |S| \leq 2^{k-1}$ be the set of all points $\mathcal{G}(\{-1,1\}^s)$. Say that $b \in \{-1,1\}^k$ is *good* if some $j \in [m]$ satisfies $\text{sign}(v_i^j) = b_i$ for all $i \in [k]$ (i.e. b is the sign-pattern of the first k coordinates of some string in S). Let $g : \{-1,1\}^k \to \{-1,1\}$ be any function which outputs -1 on each good string in $\{-1,1\}^k$ and outputs 1 on exactly 2^{k-1} strings in $\{-1,1\}^k$ (such a g must exist since $|S| \leq 2^{k-1}$ and hence there are at most 2^{k-1} good strings in $\{-1,1\}^k$). Let $h_i(x)$ be the LTF $\text{sign}(x_i)$ for each $i \in [k]$. Then for $f(x) = g(h_1(x), \ldots, h_k(x))$, we have $\mathbf{Pr}[f(\mathcal{G}(\mathbf{U}^{(s)})) = 1] = 0$ but $\mathbf{Pr}[f(\mathbf{G}^{(n)}) = 1] = 1/2$. Hence $\mathcal{G}$ cannot be a 0.49-PRG for the class of all functions of k LTFs over Gaussian space. ◁

B Simulating draws from the Gaussian distribution

In the analysis of our PRGs for arbitrary functions of k LTFs and for intersections of k LTFs, we assumed that we can sample from d-dimensional Gaussians, but to do this with perfect fidelity clearly requires infinitely many random bits. In this section we show that $O(d\log(kd/\varepsilon))$ truly random bits suffice to produce d-dimensional "approximate Gaussian" distributions that suffice for our applications.

▶ **Definition 33.** *We say that a random variable $\mathbf{G}'$ on $\mathbb{R}$ is a δ-approximate Gaussian random variable if there is a standard (correlated) Gaussian $\hat{\mathbf{G}}$ such that $\boldsymbol{Pr}[|\mathbf{G}' - \hat{\mathbf{G}}| > \delta] < \delta$.*

We recall a lemma proved by Kane [22] which generates such approximate Gaussians in a randomness efficient way. It is based on the Box-Muller transform.

▶ **Lemma 34** ([22])**.** *There is an explicit construction of a δ-approximate Gaussian random variable using $O(\log(1/\delta))$ bits of randomness.*

Let $\mathbf{G}^d$ be a $\mathcal{N}(0,1)^d$ Gaussian. Let $\hat{\mathbf{G}}^{(d)}$ denote a coordinate-wise independent distribution in which the i-th coordinate $\hat{\mathbf{G}}_i^{(d)}$ is a δ-approximate Gaussian random variable with respect to $\mathbf{G}_i^{(d)}$ as given by Lemma 34. We set (with foresight) the parameter $\delta = \varepsilon/(k\sqrt{d})$. By Lemma 34, a draw of $\hat{\mathbf{G}}^{(d)}$ can be generated using $O(d\log(kd/\varepsilon))$ bits of randomness. Below we prove that $\hat{\mathbf{G}}^{(d)}$ can be used instead of $\mathbf{G}^d$ in our PRGs, at the cost of an additional additive ε error for our PRG.

Let $\mathbf{X} = V\mathbf{G}^{(d)} - \vec{\theta}$ and $\hat{\mathbf{X}} = V\hat{\mathbf{G}}^{(d)} - \theta$. We prove that the "union-of-orthants" distance $d_{\mathrm{UO}}(\mathbf{X}, \hat{\mathbf{X}})$ between $\mathbf{X}$ and $\hat{\mathbf{X}}$ (see (18)) is at most ε. This directly implies that the approximation works since, as observed in Section 5.3, for any function $g : \{-1,1\}^k \to \{-1,1\}$, we have

$$\left|\mathbf{Pr}[g(\mathrm{sign}(\mathbf{X}_1),\ldots,\mathrm{sign}(\mathbf{X}_k)) = 1] - \mathbf{Pr}[g(\mathrm{sign}(\hat{\mathbf{X}}_1),\ldots,\mathrm{sign}(\hat{\mathbf{X}}_k)) = 1]\right| \le d_{\mathrm{UO}}(\mathbf{X}, \hat{\mathbf{X}}).$$

In order to prove that $d_{\mathrm{UO}}(\mathbf{X}, \hat{\mathbf{X}}) \le \varepsilon$, we recall some definitions from Section 5.3. Recall that

$$B_r := \left\{x \in \mathbb{R}^k : |x_i| \le r \text{ for some } i \in [k]\right\}$$

is the region of all points in $\mathbb{R}^k$ whose L^∞-distance from any orthant boundary point is at most r. Set $r = 2\delta\sqrt{d}$. For any union of orthants $\mathcal{O}$, we partition $\mathcal{O}$ into $\mathcal{O}_{\mathrm{bd}} := \mathcal{O} \cap B_r$ (the points in $\mathcal{O}$ that lie close to the orthant boundaries) and $\mathcal{O}_{\mathrm{in}} := \mathcal{O} \setminus B_r$ (the points in $\mathcal{O}$ that lie far away from the orthant boundaries).

We have

$$|\,\mathbf{Pr}[\mathbf{X} \in \mathcal{O}] - \mathbf{Pr}[\hat{\mathbf{X}} \in \mathcal{O}]| \le \mathbf{Pr}[\mathbf{X} \in \mathcal{O}_{bd}] + |\,\mathbf{Pr}[\mathbf{X} \in \mathcal{O}_{in}] - \mathbf{Pr}[\hat{\mathbf{X}} \in \mathcal{O}]|.$$

By Lemma 34 and a union bound, it follows that with probability at least $1 - k\delta$, $|\mathbf{G}_i^{(d)} - \hat{\mathbf{G}}_i^{(d)}| \le \delta$ for each $i \in [k]$. Thus, with probability at least $1 - k\delta$, for each $i \in [k]$, we have

$$|\mathbf{X}_i - \hat{\mathbf{X}}_i| = V^i \cdot (\mathbf{G}_i^{(d)} - \hat{\mathbf{G}}_i^{(d)}) \le ||V^i||_2 ||\mathbf{G}_i^{(d)} - \hat{\mathbf{G}}_i^{(d)}||_2 \le \delta\sqrt{d}.$$

As a direct consequence, we have that $\mathbf{Pr}[\hat{\mathbf{X}} \in \mathcal{O} | \mathbf{X} \notin \mathcal{O}_{in}]| \le k\delta$ and $\mathbf{Pr}[\hat{\mathbf{X}} \in \mathcal{O} | \mathbf{X} \in \mathcal{O}_{in}] \ge 1 - k\delta$. Thus,

$$\begin{aligned}\mathbf{Pr}[\hat{\mathbf{X}} \in \mathcal{O}] &\le \mathbf{Pr}[\hat{\mathbf{X}} \in \mathcal{O} | \mathbf{X} \in \mathcal{O}_{in}] \cdot \mathbf{Pr}[\mathbf{X} \in \mathcal{O}_{in}] + \mathbf{Pr}[\hat{\mathbf{X}} \in \mathcal{O} | \mathbf{X} \notin \mathcal{O}_{in}] \cdot \mathbf{Pr}[\mathbf{X} \notin \mathcal{O}_{in}] \\ &\le \mathbf{Pr}[\mathbf{X} \in \mathcal{O}_{in}] + k\delta,\end{aligned}$$

and

$$\begin{aligned}\mathbf{Pr}[\hat{\mathbf{X}} \in \mathcal{O}] &\geq \mathbf{Pr}[\hat{\mathbf{X}} \in \mathcal{O} | \mathbf{X} \in \mathcal{O}_{in}] \cdot \mathbf{Pr}[\mathbf{X} \in \mathcal{O}_{in}] \\ &\geq (1 - k\delta)\,\mathbf{Pr}[\mathbf{X} \in \mathcal{O}_{in}] \geq \mathbf{Pr}[\mathbf{X} \in \mathcal{O}_{in}] - k\delta.\end{aligned}$$

Hence, $|\,\mathbf{Pr}[\mathbf{X} \in \mathcal{O}_{in}] - \mathbf{Pr}[\hat{\mathbf{X}} \in \mathcal{O}]| \leq k\delta$.
Finally note that, as estimated in Section 5.3, using anti-concentration of Gaussians,

$$\mathbf{Pr}[\mathbf{X} \in \mathcal{O}_{bd}] \leq O(kr).$$

Combining the above estimates, we have

$$\begin{aligned}|\,\mathbf{Pr}[\mathbf{X} \in \mathcal{O}] - \mathbf{Pr}[\hat{\mathbf{X}} \in \mathcal{O}]| &\leq \mathbf{Pr}[\mathbf{X} \in \mathcal{O}_{bd}] + |\,\mathbf{Pr}[\mathbf{X} \in \mathcal{O}_{in}] - \mathbf{Pr}[\hat{\mathbf{X}} \in \mathcal{O}]| \\ &\leq O(k\delta\sqrt{d}) = O(\varepsilon),\end{aligned}$$

which concludes our proof.

C Proof of Theorem 24: An invariance principle for arbitrary functions of LTFs

C.1 Our starting point: a Wasserstein distance bound

Our proof of Theorem 24 closely parallels the arguments underlying our PRG for arbitrary functions of k LTFs that were given in Section 5. However, for technical reasons we will now be using the (non-quadratic) Wasserstein distance. We recall the definition of this distance measure between distributions that we will use. (As was the case earlier for quadratic Wasserstein distance, there is an equivalent formulation in terms of Lipschitz test functions, but we will not need this alternative formulation.)

▶ **Definition 35.** *For any two distributions* $\mathbf{X}$ *and* $\mathbf{Y}$ *over* $\mathbb{R}^k$*, the Wasserstein distance between* $\mathbf{X}$ *and* $\mathbf{Y}$ *is defined to be*

$$d_W(\mathbf{X}, \mathbf{Y}) = \inf_{(\widehat{\mathbf{X}}, \widehat{\mathbf{Y}})} (\boldsymbol{E}[\|\widehat{\mathbf{X}} - \widehat{\mathbf{Y}}\|]),$$

where the infimum is taken over all couplings $(\widehat{\mathbf{X}}, \widehat{\mathbf{Y}})$ *of* $\mathbf{X}$ *and* $\mathbf{Y}$*.*

As in the analysis of our PRG for arbitrary functions of k LTFs, we need an upper bound on the Wasserstein distance between the two random variables of interest as a starting point. In Section 5 the two relevant random variables were both multi-dimensional Gaussians and the desired (quadratic) Wasserstein closeness was given by Proposition 8. In the context of Theorem 24, the two relevant random variables are (i) a sum of independent vector-valued random variables and (ii) the Gaussian with matching mean and covariance, so it is natural to turn to the literature on *central limit theorems* for sums of vector-valued random variables for the desired upper bound on Wasserstein distance.

A range of central limit theorems for sums of independent vector-valued random variables have been established in the literature, but we are not aware of one which can be used "out of the box" for our purposes. Valiant and Valiant [32] gave a central limit theorem which upper bounds the Wasserstein distance between a sum of n vector-valued random variables and the corresponding Gaussian, but their quantitative bound has a $\log n$ factor which would spoil our desired final result. Zhai [33] gave a variant of the [32] CLT, but only for the setting of

i.i.d. vector-valued random variables, whereas our summands are not identically distributed. Bonis [3] gave a sharpening of Zhai's bound, but it assumes that each summand random variable has identity covariance, which need not hold for us. While we do not know of any CLTs in the literature which directly yield our desired starting point, below we show how a "bucketing" scheme can be applied to the Valiant-Valiant CLT to yield a CLT of exactly the type that we need (where there is no dependence on n in the upper bound).

We begin by recalling the Valiant-Valiant CLT:

▶ **Theorem 36** (Valiant-Valiant CLT for Wasserstein distance [32]). *Let $\mathbf{Z}_1, \ldots, \mathbf{Z}_n$ be independent distributions in $\mathbb{R}^k$ with mean 0 and $||\mathbf{Z}_i||_2 \leq \beta$. Then, writing Σ to denote the covariance matrix of $\mathbf{Z}_1 + \cdots + \mathbf{Z}_n$, we have*

$$d_W(\sum_{a=1}^{n} \mathbf{Z}_a, \mathcal{N}(0, \Sigma)) \leq \beta k(2.7 + 0.83 \log n).$$

We use this to prove the following:

▶ **Proposition 37.** *Let $h_1, h_2, \ldots, h_k$ be τ-regular LTFs, $h_i(x) = \mathrm{sign}(W^i_1 x_1 + \cdots + W^i_n x_n - \theta)$ where we have normalized so that each vector $W^i = (W^i_1, \ldots, W^i_n)$ has two-norm 1. Let W be the $k \times n$ matrix with (i, j) entry W^i_j, and for $\ell \in [n]$ let W_ℓ denote the column vector with entries $W^1_\ell, \ldots, W^k_\ell$. For $\ell \in [n]$ let $\mathbf{Z}_\ell$ denote the k-dimensional random variable $\mathbf{Z}_\ell = \boldsymbol{x}_\ell W_\ell$ where $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ is uniform over $\{-1, 1\}^n$ and let $\mathbf{Z} = \mathbf{Z}_1 + \cdots + \mathbf{Z}_n$. Let $\mathbf{G}'$ be the k-dimensional random Gaussian vector $\mathbf{G}' = W\mathbf{G}$ where $\mathbf{G}$ is distributed as $\mathcal{N}(0, 1)^n$. Then*

$$d_W(\mathbf{Z}, \mathbf{G}') \leq O(k^2 \log(k) \cdot \tau^2 + k). \tag{30}$$

Further, if $\tau < 10/\sqrt{k}$, then the following bound also holds:

$$d_W(\mathbf{Z}, \mathbf{G}') \leq O(k^2 \tau^2 \log(k/\tau)). \tag{31}$$

(We note that while (30) does not provide a very strong upper bound on Wasserstein distance, for suitably small values of τ the bound (31) does give a useful upper bound, and it is this bound that we will employ in the next subsection.)

Proof. We begin by observing that the random variables $\mathbf{Z}_1, \ldots, \mathbf{Z}_\ell$ are independent, have mean zero (indeed each has support size two, on the two points W_ℓ and $-W_\ell$), and lie in $\mathbb{R}^k$. However, at this point, just having the condition that the *rows* of W are τ-regular and have two-norm 1 doesn't provide much useful information about the two-norms of the columns W_ℓ. Our approach is to bucket the columns according to the two-norms and use the Valiant-Valiant CLT (Theorem 36) separately on each of these buckets. We now proceed to give more details.

Let A_i be the subset of those $\ell \in [n]$ such that $2^{-i-1} \leq \|W_\ell\|_2 \leq 2^{-i}$, i.e.

$$2^{-2i-2} \leq (W^1_\ell)^2 + \cdots + (W^k_\ell)^2 \leq 2^{-2i}.$$

Fix an $\ell \in [n]$ and consider the column vector $W_\ell = (W^1_\ell, \ldots, W^k_\ell)$. We have that each $|W^i_\ell| \leq \tau$ (using the τ-regularity of each row and the fact that each row is normalized to have 2-norm 1). Thus, we have $0 \leq (W^1_\ell)^2 + \cdots + (W^k_\ell)^2) \leq k\tau^2$. It follows that A_i is empty if $i < i_0 := (\log(1/k\tau^2))/2 - 1$. (Note that if k is large and τ is not very small then i_0 may be a negative value; this will come up below.)

The sum of squares of all $W_{i,j}$ is k, so each A_i can have at most $k \cdot 2^{2i+2} = 4k2^{2i}$ many elements. Fix an i such that A_i is nonempty (so $i \geq i_0$). Each $\ell \in A_i$ has $\|W_\ell\|_2 \leq 2^{-i}$, and hence applying the Valiant-Valiant CLT to $\sum_{\ell \in A_i} \mathbf{Z}_\ell$ (setting its parameter "β" to 2^{-i}) gives

$$\begin{aligned} d_W\left(\sum_{\ell \in A_i} \mathbf{Z}_\ell, \mathcal{N}(0, \Sigma_{(i)})\right) &\leq 2^{-i} \cdot k \cdot (2.7 + \log|A_i|) \leq 2^{-i} \cdot k \cdot (O(1) + \log k + 2i) \\ &= O(k \log(k) \cdot 2^{-i} + k \cdot i \cdot 2^{-i}). \end{aligned}$$

Now we use the fact that if $\mathbf{X}, \mathbf{Y}$ are two independent random variables and $\mathbf{U}, \mathbf{V}$ are two independent random variables, then

$$d_W(\mathbf{X} + \mathbf{Y}, \mathbf{U} + \mathbf{V}) \leq d_W(\mathbf{X}, \mathbf{U}) + d_W(\mathbf{Y}, \mathbf{V})$$

(this is easy to see from the coupling-based definition that we have given for d_W). Applying this, where the sum is over all $i \geq i_0$, since $\sum_i \sum_{\ell \in A_i} \mathbf{Z}_\ell = \mathbf{Z}$ and $\sum_i \mathcal{N}(0, \Sigma_{(i)}) = \mathbf{G}'$, we get that

$$d_W(\mathbf{Z}, \mathbf{G}') \leq \sum_{i \geq i_0} O(k \log(k) \cdot 2^{-i}) + \sum_{i \geq i_0} O(k \cdot i \cdot 2^{-i}).$$

Let us upper bound this sum, keeping in mind that $\log(1/k\tau^2)$ may be negative. The first sum is at most

$$\sum_{i \geq i_0} O(k \log(k) \cdot 2^{-i}) \leq O(k^2 \log(k) \cdot \tau^2).$$

The second sum is

$$\sum_{i \geq i_0} O(k \cdot i \cdot 2^{-i})$$

which needs to be considered with a bit of care since i_0 may be negative. Summing over any negative values of i obviously gives a negative contribution. Summing over positive values of i gives at most $O(k)$ (and we note that indeed the contribution when $i = 1$ is $\Theta(k)$). So the total sum is at most

$$O(k^2 \log(k) \cdot \tau^2 + k).$$

We note that either of the two summands may dominate depending on the relation between τ and k). However, if we assume that $\tau < 10/\sqrt{k}$ (so i_0 is a positive number), then the upper bound on the second sum above becomes $O(k^2\tau^2 \log(1/k\tau^2))$, which is at most $O(k^2\tau^2 \log(1/\tau))$, and we can bound the whole quantity by $O(k^2\tau^2 \log(k/\tau))$ as claimed. ◀

C.2 The invariance principle for arbitrary functions of LTFs

The CLT in Proposition 37 gives closeness in (non-quadratic) Wasserstein distance. As in Section 5, using arguments from [5] this can be translated into closeness in union-of-orthants distance. The details of the arguments are almost identical to the analysis from [5] since now (as in that work) one of the random variables is a sum of independent vector-valued random variables, the other is Gaussian, and the relevant Wasserstein distance under consideration is the non-quadratic Wasserstein distance. In a bit more detail, the analogue of (20) is now established, as in [5], using the Berry-Esseen theorem and the fact that each linear form is τ-regular, yielding $\Gamma \leq O(k(r + \tau))$. The upper bound on Wasserstein distance that was

provided by Theorem 7 in the [5] analysis is now provided by our Proposition 37; to be more precise, the analogue to the next-to-last centered equation in the proof of Theorem 5 of [5] in our setting is that we have $r\Delta/2 \leq d_W(\mathbf{Z}, \mathbf{G}')$ which is $O(k^2\tau^2 \log(k/\tau))$ by Proposition 37. Optimizing the choice of r to make $\Gamma + \Delta$ as small as possible, we obtain the following (we refer the reader to the proof of Theorem 5 of [5] for more details):

▶ **Theorem 38.** *Let $h_1, h_2, \ldots, h_k$ be τ-regular LTFs, $h_i(x) = \mathrm{sign}(W^i_1 x_1 + \cdots + W^i_n x_n - \theta)$ where we have normalized so that each vector $W^i = (W^i_1, \ldots, W^i_n)$ has two-norm 1. Let W be the $k \times n$ matrix with (i, j) entry W^i_j, and for $\ell \in [n]$ let W_ℓ denote the column vector with entries $W^1_\ell, \ldots, W^k_\ell$. For $\ell \in [n]$ let $\mathbf{Z}_\ell$ denote the k-dimensional random variable $\mathbf{Z}_\ell = \boldsymbol{x}_\ell W_\ell$ where $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ is uniform over $\{-1, 1\}^n$ and let $\mathbf{Z} = \mathbf{Z}_1 + \cdots + \mathbf{Z}_n$. Let $\mathbf{G}'$ be the k-dimensional random Gaussian vector $\mathbf{G}' = W\mathbf{G}$ where $\mathbf{G}$ is distributed as $\mathcal{N}(0, 1)^n$. Then*

$$d_{\mathrm{UO}}(\mathbf{Z}, \mathbf{G}') \leq O(k^{3/2}\tau\sqrt{\log(k/\tau)}).$$

(The condition $\tau < 10/\sqrt{k}$ in Proposition 37 does not necessitate any condition on τ in Theorem 38, because if $\tau \geq 10/\sqrt{k}$ then the claimed bound of Theorem 38 holds trivially.) Finally, we note that the desired invariance principle, Theorem 24, is a restatement of Theorem 38, using the connection between union-of-orthants distance and any k-variable Boolean combining function g that was formalized in Observation 12.

From DNF Compression to Sunflower Theorems via Regularity

Shachar Lovett
University of California, San Diego, CA, USA
slovett@ucsd.edu

Noam Solomon
MIT, Cambridge, MA, USA
noam.solom@gmail.com

Jiapeng Zhang
University of California, San Diego, CA, USA
jpeng.zhang@gmail.com

Abstract

The sunflower conjecture is one of the most well-known open problems in combinatorics. It has several applications in theoretical computer science, one of which is DNF compression, due to Gopalan, Meka and Reingold (Computational Complexity, 2013). In this paper, we show that improved bounds for DNF compression imply improved bounds for the sunflower conjecture, which is the reverse direction of the DNF compression result. The main approach is based on regularity of set systems and a structure-vs-pseudorandomness approach to the sunflower conjecture.

2012 ACM Subject Classification Theory of computation → Randomness, geometry and discrete structures

Keywords and phrases DNF sparsification, sunflower conjecture, regular set systems

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.5

Funding *Shachar Lovett*: Supported by NSF grant CCF-1614023.
Jiapeng Zhang: Supported by NSF grant CCF-1614023.

Acknowledgements We thank anonymous reviewers for insightful suggestions.

1 Introduction

The sunflower conjecture is one of the most well-known open problems in combinatorics. An r-sunflower is a family of r sets $S_1, \ldots, S_r$ where all pairwise intersections are the same. A w-set system is a collection of sets where each set has size at most w. Erdős and Rado [3] asked how large can a w-set system be, without containing an r-sunflower. They proved an upper bound of $w!(r-1)^w$, and conjectured that the bound can be improved.

▶ **Conjecture 1** (Sunflower conjecture, [3]). *Let $r \geq 3$. There is a constant c_r such that any w-set system $\mathcal{F}$ of size $|\mathcal{F}| \geq c_r^w$ contains an r-sunflower.*

60 years later, only lower order improvements have been achieved, and the best bounds are still of the order of magnitude of about w^w for any fixed r, same as in the original theorem of Erdős and Rado. A good survey on the current bounds is [6].

Sunflowers have been useful in various areas in theoretical computer science. Some examples include monotone circuit lower bounds [9, 10], barriers for improved algorithms for matrix multiplication [1] and faster deterministic counting algorithms via DNF compression [5]. The focus on this paper is on this latter application, in particular DNF compression.

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka; Article No. 5; pp. 5:1–5:14

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

A DNF (Disjunctive Normal Form) is disjunction of conjunctive terms. The *size* of a DNF is the number of terms, and the *width* of a DNF is the maximal number of literals in a term. It is a folklore result that any DNF of size s can be approximated by another DNF of width $O(\log s)$, by removing all terms of larger width. The more interesting direction is whether DNFs of small width can be approximated by DNFs of small size. Namely - can DNFs of small width be "compressed" while approximately preserving their computational structure?

A beautiful result of Gopalan, Meka and Reingold [5] shows that DNFs of small width can be approximated by small size DNFs. Their proof relies on the sunflower theorem (more precisely, a variant thereof due to Rossman [10] that we will discuss shortly). Before stating their result, we introduce some necessary terminology. We say that two functions $f, g : \{0,1\}^n \to \{0,1\}$ are ε-close if $\Pr[f(x) \neq g(x)] \leq \varepsilon$ over a uniformly chosen input. We say that f is a lower bound of g, or that g is an upper bound of f, if $f(x) \leq g(x)$ for all x.

▶ **Theorem 2** (DNF compression using sunflowers, sandwiching bounds [5]). *Let f be a width-w DNF. Then for every $\varepsilon > 0$ there exist two width-w DNFs, f_{lower} and f_{upper} such that*

(i) *$f_{lower}(x) \leq f(x) \leq f_{upper}(x)$ for all x.*
(ii) *f_{lower} and f_{upper} are ε-close.*
(iii) *f_{lower} and f_{upper} have size $(w \log(1/\varepsilon))^{O(w)}$.*

Recently, Lovett and Zhang [8] improved the dependence of the size of the lower bound DNF on w (but with a worse dependence on ε). In particular, the proof avoids the use of the sunflower theorem.

▶ **Theorem 3** (DNF compression without sunflowers, lower bound [8]). *Let f be a width-w DNF. Then for every $\varepsilon > 0$ there exists a width-w DNFs f_{lower} such that*

(i) *$f_{lower}(x) \leq f(x)$ for all x.*
(ii) *f_{lower} and f are ε-close.*
(iii) *f_{lower} has size $(1/\varepsilon)^{O(w)}$.*

It is natural to speculate that a similar bound holds for upper bound DNFs.

▶ **Conjecture 4** (Improved upper bound DNF compression). *Let f be a width-w DNF. Then for every $\varepsilon > 0$ there exists a width-w DNF f_{upper} such that*

(i) *$f(x) \leq f_{upper}(x)$ for all x.*
(ii) *f_{upper} and f are ε-close.*
(iii) *f_{upper} has size $(1/\varepsilon)^{O(w)}$.*

To study the connection between DNF compression and sunflowers, we would need an analog of Conjecture 4 for monotone DNFs.

▶ **Conjecture 5** (Improved upper bound monotone DNF compression). *In Conjecture 4, if f is a monotone DNF, then f_{upper} can also be taken to be a monotone DNF.*

The main result of this paper is that Conjecture 5 implies an improved bound for the sunflower conjecture, with a bound of $(\log w)^{O(w)}$ instead of the current bound of $O(w)^w$. Thus, the connection between sunflower theorems and DNF compression goes both ways. We note that the proof of [5] is also true for monotone DNF compression.

To simplify the presentation, we assume from now on that $w \geq 2$. This will allow us to assume that $\log w > 0$. In any case, for $w = 1$ the sunflower conjecture is trivial, as any 1-set system of size r is an r-sunflower.

▶ **Theorem 6** (Main theorem). *Assume that Conjecture 5 holds. Then for any $r \geq 3$ there exists a constant c_r such that the following holds. Any w-set system $\mathcal{F}$ of size $|\mathcal{F}| \geq (\log w)^{c_r w}$ contains an r-sunflower.*

In fact, Theorem 6 holds even with a slightly weaker conjecture instead of Conjecture 5, where the size bound can be assumed to be $((\log w)/\varepsilon)^{O(w)}$ instead of $(1/\varepsilon)^{O(w)}$.

1.1 Proof overview

The proof of Erdős and Rado [3] is by a simple case analysis which we now recall. Let $\mathcal{F}$ be a w-set system. Then either $\mathcal{F}$ contains r disjoint sets, which are in particular an r-sunflower; or at most $r-1$ sets whose union intersects all other sets. In the latter case, there is an element that belongs to a $\frac{1}{(r-1)w}$ fraction of the sets in $\mathcal{F}$. If we restrict to these sets, and remove the common element, then we reduced the problem to a $(w-1)$-set system of size $\frac{|\mathcal{F}|}{(r-1)w}$. The proof concludes by induction.

Our approach is to refine this via a structure-vs-pseudorandomness approach. Either there is a set T of elements that belong to many sets in $\mathcal{F}$ (concretely, at least $|\mathcal{F}|/\kappa^{|T|}$, for an appropriately chosen κ), or otherwise the set system $\mathcal{F}$ is pseudo-random, in the sense that no set T is contained in too many sets in $\mathcal{F}$. The main challenge is showing that by choosing κ large enough, this notion of pseudo-randomness is useful. This will involve introducing several new concepts and tying them to the sunflower problem.

The following proof overview follows the same structure as the sections in the paper, to ease readability.

Section 2: DNFs and set systems

First, we note that set systems are one-to-one correspondence to monotone DNFs. Formally, we identify a set system $\mathcal{F} = \{S_1, \ldots, S_m\}$ with the monotone DNF $f_{\mathcal{F}}(x) = \bigvee_{S \in \mathcal{F}} \bigwedge_{i \in S} x_i$. This equivalence will be useful in the proof, as at different stages one of these viewpoints is more convenient.

The notions of "lower bound DNF" f_{lower} and "upper bound DNF" f_{upper} used in Theorem 2, Theorem 3 and Conjecture 5 have analogs for set systems, which we refer to as *proper* lower bound and upper bound DNFs (or set systems). For the purpose of this high level overview, we ignore this distinction here.

Section 3: Approximate sunflowers

The notion of approximate sunflowers was introduced by Rossman [10]. It relies on the notion of *satisfying set systems*.

Let $\mathcal{F}$ be a set system on a universe X. We say that $\mathcal{F}$ is (p, ε)-*satisfying* if $\Pr_{x \sim X_p}[f_{\mathcal{F}}(x) = 1] > 1 - \varepsilon$, where $f_{\mathcal{F}}$ is the corresponding monotone DNF for $\mathcal{F}$, and X_p is the p-biased distribution on X. The importance of satisfying set systems in our context is that a $(1/r, 1/r)$-satisfying set system contains r disjoint sets (Claim 14).

Let $K = \cap_{S \in \mathcal{F}} S$ be the intersection of all sets in $\mathcal{F}$. We say that $\mathcal{F}$ is a (p, ε)-*approximate sunflower* if the set system $\{S \setminus K : S \in \mathcal{F}\}$ is (p, ε)-satisfying. An interesting connection between approximate sunflowers and sunflowers is that a $(1/r, 1/r)$-approximate sunflower contains an r-sunflower (Corollary 15).

Section 4: Regular set systems

Let $\mathcal{D}$ be a distribution over subsets of X. We say that $\mathcal{D}$ is *regular* if when sampling $S \sim \mathcal{D}$, the probability that S contains any given set T is exponentially small in the size of T. Formally, $\mathcal{D}$ is κ-regular if for any set $T \subseteq X$ it holds that $\Pr_{S\sim\mathcal{D}}[T \subseteq S] \leq \kappa^{-|T|}$.

A set system $\mathcal{F}$ is κ-regular if there exists a κ-regular distribution supported on sets in $\mathcal{F}$. We show that if $\mathcal{F}$ is κ-regular, then the same holds for any upper bound set system (Claim 18) and any "large enough" lower bound set system (Claim 19). These facts will turn out to be useful later.

Section 5: Regular set systems are $(1/2, 1/2)$-satisfying

In this section, we focus on regular set systems $\mathcal{F}$, or equivalently regular DNFs $f = f_{\mathcal{F}}$. We show that, assuming Conjecture 5 (or the slightly weaker Conjecture 21), any κ-regular DNF of width w, where $\kappa = (\log w)^{O(1)}$, is $(1/2, 1/2)$-satisfying. Namely, $\Pr[f(x) = 1] \geq 1/2$, where x is uniformly chosen. In particular, this implies that $\mathcal{F}$ contains two disjoint sets. However, our goal is to prove that $\mathcal{F}$ contains an r-sunflower for $r \geq 3$, so we are not done yet.

Section 6: Intersecting regular set systems

Let $\alpha(w, r)$ denote the maximal κ such that there exists a κ-regular w-set system without r disjoint sets. It is easy to prove that the sunflower theorem holds for any set system of size $|\mathcal{F}| > \alpha(w, r)^w$ (Claim 29). However, our discussion so far only allows us to bound $\beta(w) = \alpha(w, 2)$; concretely, assuming Conjecture 5 we have $\beta(w) \leq (\log w)^{O(1)}$.

We show (Lemma 30) that nontrivial upper bounds on $\beta(w)$ imply related upper bounds on $\alpha(w, r)$ for every r. Concretely, if $\beta(w) \leq (\log w)^{O(1)}$ then $\alpha(w, r) \leq (\log w)^{c_r}$ where $c_r > 0$ are constants. This concludes the proof, as we get that any w-set system of size $|\mathcal{F}| \geq (\log w)^{c_r w}$ must contain an r-sunflower.

2 DNFs and set systems

A DNF is *monotone* if it contains no negated variables. Monotone DNFs are in one-to-one correspondence with set systems. Formally, if $\mathcal{F}$ is a set system then the corresponding monotone DNF is

$$f_{\mathcal{F}}(x) = \bigvee_{S\in\mathcal{F}} \bigwedge_{i\in S} x_i.$$

In the other direction, if $f = \bigvee_{j\in[m]} \bigwedge_{i\in S_j} x_i$ is a monotone DNF then its corresponding set system is

$$\mathcal{F}_f = \{S_1, \ldots, S_m\}.$$

Observe that a w-set system corresponds to a width-w monotone DNF, and vice versa. If X is the set of elements over which $\mathcal{F}$ is defined then we write $\mathcal{F} \subseteq \mathcal{P}(X)$.

To recall, we consider both lower bound and upper bound DNFs. As our main motivation is to better understand sunflowers, we restrict attention to monotone DNFs from now on; however, all the definitions can be easily adapted for general DNFs.

We next define *proper* upper and lower bound DNFs. Proper lower bound DNFs are obtained by removing terms from the DNF, and proper upper bound DNFs are obtained by removing variables from terms in the DNF. We describe both in terms of the corresponding set systems.

▶ **Definition 7** (Proper lower bound DNF / set system). *Let $\mathcal{F}$ be a set system. A* proper lower bound set system *for $\mathcal{F}$ is simply a sub set system $\mathcal{F}' \subseteq \mathcal{F}$. Observe that indeed*

$$f_{\mathcal{F}'}(x) \leq f_{\mathcal{F}}(x) \qquad \forall x.$$

▶ **Definition 8** (Proper upper bound DNF / set system). *Let $\mathcal{F}$ be a set system. A* proper upper bound set system *for $\mathcal{F}$ is a set system $\mathcal{F}'$ that satisfies the following: for each $S \in \mathcal{F}$ there exists $S' \in \mathcal{F}'$ such that $S' \subseteq S$. Observe that indeed*

$$f_{\mathcal{F}'}(x) \geq f_{\mathcal{F}}(x) \qquad \forall x.$$

For monotone DNFs, upper bounds and proper upper bounds are the same.

▷ Claim 9. Let $\mathcal{F}, \mathcal{F}'$ be set systems over the same universe, such that

$$f_{\mathcal{F}'}(x) \geq f_{\mathcal{F}}(x) \qquad \forall x.$$

Then $\mathcal{F}'$ is a proper upper bound set for $\mathcal{F}$.

Proof. Assume not. Then there exists $S \in \mathcal{F}$ such that there is no $S' \in \mathcal{F}'$ with $S' \subseteq S$. Let $x = 1_S$ be the indicator vector for S. Then $f_{\mathcal{F}}(x) = 1$ but $f_{\mathcal{F}'}(x) = 0$, a contradiction. ◁

▶ **Corollary 10.** *In Conjecture 5, we may assume that f_{upper} is a proper upper bound DNF for f.*

We note that the lower and upper bound DNFs in [5] are in fact proper lower and upper bounds, and the same holds for the lower bound DNF in [8].

3 Approximate sunflowers

We introduce the notion of *approximate sunflowers*, first defined by Rossman [10]. We first need some notation. Given a finite set X and $0 < p < 1$, we denote by X_p the *p-biased* distribution over X, where $W \sim X_p$ is sampled by including each $x \in X$ in W independently with probability p. The definition of approximate sunflowers relies on the notion of a *satisfying set system.*

▶ **Definition 11** (Satisfying set system). *Let $\mathcal{F} \subseteq \mathcal{P}(X)$ be a set system and let $0 < p, \varepsilon < 1$. We say that $\mathcal{F}$ is (p, ε)-satisfying if*

$$\Pr_{W \sim X_p}[\exists S \in \mathcal{F} : S \subseteq W] > 1 - \varepsilon.$$

Equivalently, if $f_{\mathcal{F}} : \{0,1\}^X \to \{0,1\}$ is the DNF corresponding to $\mathcal{F}$, then $\mathcal{F}$ is (p, ε)-satisfying if

$$\Pr_{x \sim X_p}[f_{\mathcal{F}}(x) = 1] > 1 - \varepsilon.$$

An approximate sunflower is a set system which is satisfying if we first remove the common intersection of all the sets in the set system.

▶ **Definition 12** (Approximate sunflower). *Let $\mathcal{F} \subseteq \mathcal{P}(X)$ be a set system and let $0 < p, \varepsilon < 1$. Let $K = \cap_{S \in \mathcal{F}} S$. Then $\mathcal{F}$ is a (p, ε)-approximate sunflower if the set system $\{S \setminus K : S \in \mathcal{F}\}$ is (p, ε)-satisfying.*

Rossman proved an analog of the sunflower theorem for approximate sunflowers. Li, Lovett and Zhang [7] reproved this theorem by using a connection to randomness extractors.

▶ **Theorem 13** (Approximate sunflower lemma [10]). *Let $\mathcal{F}$ be a w-set system and let $\varepsilon > 0$. If $|\mathcal{F}| \geq w! \cdot (1.71 \log(1/\varepsilon)/p)^w$ then $\mathcal{F}$ contains a (p, ε)-approximate sunflower.*

To conclude this section, we show that satisfying set systems contain many disjoint sets, and hence approximate sunflowers contain sunflowers.

▷ Claim 14. Let $r \geq 2$ and $\mathcal{F}$ be a $(1/r, 1/r)$-satisfying set system. Then $\mathcal{F}$ contains r pairwise disjoint sets.

Proof. Let $\mathcal{F} \subseteq \mathcal{P}(X)$. Consider a uniform random coloring of X with r colors. A coloring induces a partition of X into $X = W_1 \cup \ldots \cup W_r$, where W_c is the set of all elements that attain the color c. Given a color $c \in [r]$, a set $S \in \mathcal{F}$ is c-monochromatic if all its elements attain the color c. Observe that for each color c,

$$\Pr[\exists S \in \mathcal{F},\ S \text{ is } c\text{-monochromatic}] = \Pr[\exists S \in \mathcal{F}, S \subseteq W_c].$$

The marginal distribution of each W_c is $(1/r)$-biased. By our assumption that $\mathcal{F}$ is $(1/r, 1/r)$-satsifying, the probability that W_c contains some $S \in \mathcal{F}$ is more than $1 - 1/r$. So by the union bound,

$$\Pr[\forall c \in [r]\ \exists S \in \mathcal{F},\ S \text{ is } c\text{-monochromatic}] > 0.$$

In particular, there exists a coloring where this event happens. Let $S_1, \ldots, S_r$ be the sets for which S_c is c-monochromatic. Then $S_1, \ldots, S_r$ must be pairwise disjoint. ◁

▶ **Corollary 15.** *Let $\mathcal{F}$ be a $(1/r, 1/r)$-approximate sunflower. Then $\mathcal{F}$ contains an r-sunflower.*

Proof. Let $K = \cap_{S \in \mathcal{F}} S$. Apply Claim 14 to the set system $\mathcal{F}' = \{S \setminus K : S \in \mathcal{F}\}$ which by assumption is $(1/r, 1/r)$-satisfying. We obtain that $\mathcal{F}'$ contains r pairwise disjoint sets $S_1 \setminus K, \ldots, S_r \setminus K$. This implies that $S_1, \ldots, S_r$ form an r-sunflower. ◀

4 Regular set systems

The notion of regularity of a set system is pivotal in this paper. At a high level, a set system is regular if no element belongs to too many sets, no pair of elements belongs to too many sets, and so on. It is closely related to the notion of block min-entropy studied in the context of lifting theorems in communication complexity [4].

▶ **Definition 16** (Regular distribution). *Let X be a finite set, and let $\mathcal{D}$ be a distribution on subsets $S \subseteq X$. The distribution $\mathcal{D}$ is κ-regular if for any set $T \subseteq X$ it holds that*

$$\Pr_{S \sim \mathcal{D}}[T \subseteq S] \leq \kappa^{-|T|}.$$

▶ **Definition 17** (Regular set system). *A set system $\mathcal{F}$ is κ-regular if there exists a κ-regular distribution $\mathcal{D}$ supported on the sets in $\mathcal{F}$.*

The following claims show that if $\mathcal{F}$ is a κ-regular set system then any proper upper bound set system for it is also κ-regular, and any "large" proper lower bound set system is approximately κ-regular.

▷ Claim 18. Let $\mathcal{F}$ be a κ-regular set system. Let $\mathcal{F}'$ be a proper upper bound set system for $\mathcal{F}$. Then $\mathcal{F}'$ is also κ-regular.

Proof. Let $\mathcal{D}$ be a κ-regular distribution supported on $\mathcal{F}$. Let $\varphi : \mathcal{F} \to \mathcal{F}'$ be a map such that $\varphi(S) \subseteq S$ for all $S \in \mathcal{F}$. Define a distribution $\mathcal{D}'$ on $\mathcal{F}'$ as follows:

$$\mathcal{D}'(S') = \sum_{S \in \varphi^{-1}(S')} \mathcal{D}(S).$$

Then for any set T,

$$\Pr_{S' \sim \mathcal{D}'}[T \subseteq S'] = \sum_{S' \in \mathcal{F}' : T \subseteq S'} \mathcal{D}'(S') = \sum_{S \in \mathcal{F} : T \subseteq \varphi(S)} \mathcal{D}(S) \leq \sum_{S \in \mathcal{F} : T \subseteq S} \mathcal{D}(S) = \Pr_{S \sim \mathcal{D}}[T \subseteq S]$$

Since $\mathcal{D}$ is a regular distribution, the claim then follows. ◁

▷ Claim 19. Let $\mathcal{F}$ be a κ-regular set system, and $\mathcal{D}$ be a κ-regular distribution supported on $\mathcal{F}$. Let $\mathcal{F}' \subseteq \mathcal{F}$ be a proper lower bound set system for $\mathcal{F}$, and let $\alpha = \mathcal{D}(\mathcal{F}')$. Then $\mathcal{F}'$ is $(\kappa\alpha)$-regular.

Proof. Define a distribution $\mathcal{D}'$ on $\mathcal{F}'$ by $\mathcal{D}'(S) = \alpha^{-1}\mathcal{D}(S)$. Then for any non-empty set T,

$$\Pr_{S \sim \mathcal{D}'}[T \subseteq S] \leq \alpha^{-1} \Pr_{S \sim \mathcal{D}}[T \subseteq S] \leq \alpha^{-1}\kappa^{-|T|} \leq (\kappa\alpha)^{-|T|}.$$ ◀

5 Regular set systems are $(1/2, 1/2)$-satisfying

In this section we use Conjecture 5 to prove that regular enough DNFs are $(1/2, \varepsilon)$-satisfying, where in light of Claim 14 we care about $\varepsilon = 1/2$. To recall the definitions, a DNF f is $(1/2, \varepsilon)$-satisfying if for a uniformly chosen x,

$$\Pr_x[f(x) = 1] > 1 - \varepsilon.$$

Define

$$\gamma(w) = \sup\{\kappa : \exists \kappa\text{-regular } w\text{-set system which is } \underline{\text{not}} \ (1/2, 1/2)\text{-satisfying}\}.$$

We start by giving a lower bound on $\gamma(w)$, where the motivation is to help the reader gain intuition.

▷ Claim 20. $\gamma(w) \geq \log w - O(1)$.

Proof. We construct a κ-regular w-set system which is not $(1/2, 1/2)$-satisfying, for $\kappa = \log w - O(1)$. Let $X_1, \ldots, X_w$ be disjoint sets, each of size $\kappa = \log w - c$ for a constant $c > 0$ to be determined. Let $X = X_1 \cup \ldots \cup X_w$. Let $\mathcal{F} \subseteq \mathcal{P}(X)$ be the w-set system of all sets S that contain exactly one element from each set X_i. It is simple to verify that the uniform distribution over $\mathcal{F}$ is κ-regular, and hence $\mathcal{F}$ is κ-regular. Let $W \sim X_{1/2}$. Then

$$\Pr[\exists S \in \mathcal{F},\ S \subseteq W] = \Pr[\forall i \in [w], |X_i \cap W| \geq 1] = (1 - 2^{-\kappa})^w = (1 - c/w)^w \leq \exp(-c).$$

In particular, for $c \geq 1$ we get that $\mathcal{F}$ is not $(1/2, 1/2)$-satisfying. ◁

As we shall soon see, Conjecture 5 implies that the lower bound is not far from tight:

$$\gamma(w) \leq (\log w)^{O(1)}.$$

It will be sufficient to assume a slightly weaker version of Conjecture 5, where we allow the size of f_{upper} to be somewhat bigger.

▶ **Conjecture 21** (Weaker version of Conjecture 5)**.** *Let $w \geq 2, \varepsilon > 0$. For any monotone width-w DNF f there exists a monotone width-w DNF f_{upper} such that*

(i) *f_{upper} is a proper upper bound DNF for f.*
(ii) *f_{upper} and f are ε-close.*
(iii) *f_{upper} has size at most $((\log w)/\varepsilon)^{cw}$ for some absolute constant $c > 1$.*

▶ **Lemma 22.** *Assume Conjecture 21 holds. Then there exists a constant $c_0 > 1$ such that the following holds. For $w \geq 2, \varepsilon > 0$ let $\kappa_0(w, \varepsilon) = ((\log w)/\varepsilon)^{c_0}$. Let $\mathcal{F}$ be a w-set system which is $\kappa_0(w, \varepsilon)$-regular. Then $\mathcal{F}$ is $(1/2, \varepsilon)$-satisfying.*

▶ **Corollary 23.** $\gamma(w) \leq \kappa_0(w, 1/2) = (\log w)^{O(1)}$.

We prove Lemma 22 in the remainder of this section. We start with some simple claims that would serve as a base case for Lemma 22 for $w = O(1)$.

▷ Claim 24. Let $r \geq 2$. Let $\mathcal{F}$ be a κ-regular w-set system, where $\kappa > w\binom{r}{2}$. Then $\mathcal{F}$ contains r pairwise disjoint sets.

Proof. Let $\mathcal{D}$ be a κ-regular distribution over $\mathcal{F}$. Sample independently $S, S' \sim \mathcal{D}$. The probability that S, S' intersect is at most

$$\Pr[|S \cap S'| \geq 1] \leq \sum_{i \in S} \Pr[i \in S'] \leq w/\kappa.$$

Let $S_1, \ldots, S_r \sim \mathcal{D}$ be chosen independently. Then by the union bound, the probability that two of them intersect is at most $\binom{r}{2} w/\kappa < 1$. In particular, there exist r pairwise disjoint sets in $\mathcal{F}$. ◁

▷ Claim 25. Let $\varepsilon > 0$. Let $\mathcal{F}$ be a κ-regular w-set system, where $\kappa = w(2^w \log(1/\varepsilon))^2$. Then $\mathcal{F}$ is $(1/2, \varepsilon)$-satisfying.

Proof. Assume $\mathcal{F} \subseteq \mathcal{P}(X)$. Claim 24 implies that $\mathcal{F}$ contains $r = 2^w \log(1/\varepsilon)$ disjoint sets $S_1, \ldots, S_r$. Let $W \sim X_{1/2}$. Then

$$\Pr[\exists S \in \mathcal{F}, S \subseteq W] \geq \Pr[\exists i \in [r], S_i \subseteq W] = 1 - (1 - 2^{-w})^r > 1 - \varepsilon.$$ ◀

Proof of Lemma 22. We will need several properties from κ_0 in the proof. To simplify notations, we shorthand $\kappa_0(w/2, \varepsilon)$ for $\kappa_0(\lfloor w/2 \rfloor, \varepsilon)$ throughout. The constant $c > 1$ below is the absolute constant from Conjecture 5. We need a constant $c' > 1$ so that the following conditions are satisfied:

(i) $\kappa_0(w, \varepsilon) \geq w(2^w \log(1/\varepsilon))^2$ for $w = 1, 2$ and $\varepsilon > 0$.
(ii) $\kappa_0(w, \varepsilon) \geq ((\log w)/\varepsilon)^{12c}$ for $w \geq 3, \varepsilon > 0$.
(iii) $\kappa_0(w, \varepsilon) \geq \kappa_0(w/2, \varepsilon(1 - 1/\log w)) + 1$ for $w \geq 3, \varepsilon > 0$.

One can check that the function $\tau(w, \varepsilon) = (\log w)/\varepsilon$ satisfies $\tau(w/2, \varepsilon(1 - 1/\log w)) \leq \tau(w, \varepsilon)$, with equality when w is even. Thus taking $\kappa_0(w, \varepsilon) = ((\log w)^2/\varepsilon)^{c'}$ satisfies the conditions for a large enough $c' \geq 12c$. We then take $c_0 = 2c'$.

The proof of lemma Lemma 22 is by induction on w. The base cases are $w = 1$ and $w = 2$ which follow from Claim 25 and condition (i) on κ_0. Thus, we assume from now that $w \geq 3$. We need to prove that for $f = f_{\mathcal{F}}$ we have

$$\Pr[f(x) = 0] < \varepsilon.$$

Let $\gamma = \varepsilon/\log w$ and assume that $\mathcal{F}$ is κ-regular for $\kappa = \kappa_0(w, \varepsilon)$. Let $\mathcal{F}_1 = \{S \in \mathcal{F} : |S| \geq w/2\}$ and let $f_1 = f_{\mathcal{F}_1}$ be the corresponding DNF. Applying Conjecture 21 to f_1 with error parameter γ, we obtain that there exists a γ-approximate proper upper bound DNF f_2 for f_1 of size $s = ((\log w)/\gamma)^{cw} \leq ((\log w)/\varepsilon)^{2cw}$. Let $\mathcal{F}_2$ be the corresponding set system to f_2, and observe that $\mathcal{F}_2$ is a proper upper bound set system for $\mathcal{F}_1$. Let $\mathcal{F}_3 = (\mathcal{F} \setminus \mathcal{F}_1) \cup \mathcal{F}_2$ and let $f_3 = f_{\mathcal{F}_3}$ be the corresponding DNF. Then

$$\Pr[f(x) = 0] \leq \Pr[f_3(x) = 0] + (\Pr[f_2(x) = 0] - \Pr[f_1(x) = 0]) \leq \Pr[f_3(x) = 0] + \gamma.$$

Next, observe that $\mathcal{F}_3$ is a proper upper bound set system for $\mathcal{F}$. As we assume that $\mathcal{F}$ is κ-regular, then by Claim 18 we obtain that $\mathcal{F}_3$ is also κ-regular. Let $\mathcal{D}$ be a κ-regular distribution supported on $\mathcal{F}_3$. Let $\mathcal{F}_4 = \{S \in \mathcal{F}_3 : |S| \geq w/2\}$, where $\mathcal{F}_4 \subseteq \mathcal{F}_2$. As each set $S \in \mathcal{F}_4$ has size $|S| \geq w/2$ then, since D is κ-regular, we have

$$\mathcal{D}(S) \leq \kappa^{-w/2}.$$

Summing over all $S \in \mathcal{F}_4$ we obtain that

$$\mathcal{D}(\mathcal{F}_4) \leq |\mathcal{F}_4| \cdot \kappa^{-w/2} \leq |\mathcal{F}_2| \cdot \kappa^{-w/2} \leq \left(\left(\frac{\log w}{\varepsilon}\right)^{2c} \kappa^{-1/2}\right)^w.$$

We would need that $\mathcal{D}(\mathcal{F}_4) \leq 1/\kappa$. As $w \geq 3$, this follows from condition (ii) on κ_0. Let $\mathcal{F}_5 = \mathcal{F}_3 \setminus \mathcal{F}_4$. Then $\mathcal{F}_5$ is a $(w/2)$-set system. By Claim 19 $\mathcal{F}_5$ is κ'-regular for

$$\kappa' = \kappa \cdot \mathcal{D}(\mathcal{F}_5) = \kappa(1 - \mathcal{D}(\mathcal{F}_4)) \geq \kappa - 1.$$

Let $\varepsilon' = \varepsilon(1 - 1/\log w)$. Assumption (iii) on κ_0 gives that $\kappa_0(w/2, \varepsilon') \leq \kappa_0(w, \varepsilon) - 1$. Thus, $\mathcal{F}_5$ is $\kappa_0(w/2, \varepsilon')$-regular. Applying the induction hypothesis, if we denote by f_5 the corresponding DNF for $\mathcal{F}_5$, then

$$\Pr[f_5 = 0] < \varepsilon'.$$

Finally, as $\mathcal{F}_5 \subseteq \mathcal{F}_3$ we have $\Pr[f_3(x) = 0] \leq \Pr[f_5(x) = 0]$. Putting these together we obtain that

$$\begin{aligned}\Pr[f(x) = 0] &\leq \Pr[f_3(x) = 0] + \gamma \\ &\leq \Pr[f_5(x) = 0] + \gamma < \varepsilon' + \gamma = \varepsilon(1 - 1/\log w) + \varepsilon/\log w = \varepsilon.\end{aligned}$$

◀

6 Intersecting regular set systems

As we showed in Claim 14, if $\mathcal{F}$ is a $(1/r, 1/r)$-satisfying set system, then it contains an r-sunflower. However we only proved that a regular enough set system is $(1/2, 1/2)$-satisfying so far. In this section, we prove that this is enough to show the existence of an r-sunflower for any constant r, and with a comparable condition of regularity. Our proof is based on a the study of regular intersecting set systems.

▶ **Definition 26** (Intersecting set system)**.** *A set system is intersecting if any two sets in it intersect. In other words, it does not contain two disjoint sets.*

▶ **Definition 27.** *For $w \geq 1, r \geq 2$ define*

$$\alpha(w, r) = \sup\{\kappa : \exists \kappa\text{-regular } w\text{-set system without } r \text{ pairwise disjoint sets}\}.$$

It will be convenient to shorthand $\beta(w) = \alpha(w, 2)$, which can equivalently be defined as

$$\beta(w) = \sup\{\kappa : \exists \kappa\text{-regular intersecting } w\text{-set system}\}.$$

▷ Claim 28. $\alpha(w+1,r) \geq \alpha(w,r)$ and $\alpha(w,r+1) \geq \alpha(w,r)$ for all $w \geq 1, r \geq 2$.

Proof. The first claim follows by our definition that a w-set system is a set system where all sets have size at most w. In particular, any w-set system is also a $(w+1)$-set system and hence $\alpha(w+1,r) \geq \alpha(w,r)$. The second claim holds since a set system that does not contain r disjoint sets, also does not contain $r+1$ disjoint sets. ◁

We start by showing that upper bounds on $\alpha(w,r)$ directly translate to upper bounds on sunflowers. This is reminiscent to the original proof of Erdős and Rado [3].

▷ Claim 29. Let $\mathcal{F}$ be a w-set system of size $|\mathcal{F}| > \alpha(w,r)^w$. Then $\mathcal{F}$ contains an r-sunflower.

Proof. The proof is by induction on w. If $\mathcal{F}$ contains r pairwise disjoint sets then we are done. Otherwise, $\mathcal{F}$ is not κ-regular for any $\kappa > \alpha(w,r)$. In particular, the uniform distribution over $\mathcal{F}$ is not κ-regular. This implies that there exists a nonempty set T of size $|T| = t \geq 1$ such that

$$\mathcal{F}' = \{S \setminus T : S \in \mathcal{F}, T \subseteq S\}$$

has size $|\mathcal{F}'| \geq |\mathcal{F}|\kappa^{-t} > \alpha(w,r)^{w-t} \geq \alpha(w-t,r)^{w-t}$. By induction, $\mathcal{F}'$ contains an r-sunflower $S_1 \setminus T, \ldots, S_r \setminus T$. Hence $S_1, \ldots, S_r$ is a sunflower in $\mathcal{F}$. ◁

The main lemma we prove in this section is that upper bounds on β imply upper bounds on α.

▶ **Lemma 30.** *For all $w \geq 1, r \geq 3$ it holds that $\alpha(w,r) \leq r2^{r+1}\beta(wr)^r$.*

Before proving Lemma 30, we first prove some upper and lower bounds on $\beta(w)$. Although these are not needed in the proof of Lemma 30, we feel that they help gain intuition on $\beta(w)$.

▷ Claim 31. $\beta(w) \leq w$.

Proof. Apply Claim 24 for $r = 2$. ◁

It is easy to construct examples that show that $\beta(w) > 1$; for example, the family of all sets of size w in a universe of size $2w-1$ is intersecting and $((2w-1)/w)$-regular. The following example shows that $\beta(w)$ is super-constant.

▷ Claim 32. $\beta(w) \geq \Omega\left(\frac{\log w}{\log\log w}\right)$.

Proof. We construct an example of an intersecting w-set system for $\kappa = \Omega(\log w/\log\log w)$. Let $t \leq w/2$ to be optimized later and set $m = w - t + 1$. Let $X_1, \ldots, X_m$ be disjoint sets of size t each, and let $X = X_1 \cup \ldots \cup X_m$. Consider the set system $\mathcal{F}$ of all sets $S \subseteq X$ of the following form:

$$\mathcal{F} = \{S \subseteq X : \exists i \in [m],\ X_i \subseteq S,\ \forall j \neq i,\ |X_j \cap S| = 1\}.$$

Observe that $\mathcal{F}$ is an intersecting w-set system.

Let $\mathcal{D}$ be the uniform distribution over $\mathcal{F}$. We show that $\mathcal{D}$ is κ-regular, and hence $\mathcal{F}$ is κ-regular. There are two extreme cases: for sets T of size $|T| = 1$ we have

$$\Pr_{S\sim\mathcal{D}}[T \subseteq S] = \frac{1}{m} + \left(1 - \frac{1}{m}\right)\frac{1}{t} \leq \frac{2}{t}.$$

For sets $T = X_i$ we have

$$\Pr_{S \sim \mathcal{D}}[X_i \subseteq S] = \frac{1}{m}.$$

One can verify that these are the two extreme cases which control the regularity, and hence $\mathcal{F}$ is κ-regular for

$$\kappa = \min(t/2, m^{1/t}).$$

Setting $t = \Theta(\log w / \log\log w)$ gives $\kappa = \Theta(\log w / \log\log w)$. ◁

We conjecture that this is essentially tight. In fact, by Claim 14 we have that

$$\beta(w) \leq \gamma(w)$$

As we proved, Conjecture 21 implies $\gamma(w) = (\log w)^{O(1)}$, thus it also implies $\beta(w) = (\log w)^{O(1)}$.

Proof of Lemma 30. For $w, r \geq 1$ define

$$\eta(w, r) = r 2^{r+1} \beta(wr)^r.$$

We will first prove that

$$\alpha(w, 2r) \leq \max\left(\eta(w, r), 2\alpha(w, r)\right)$$

and then that this implies the bound

$$\alpha(w, r) \leq \eta(w, r).$$

Let $\mathcal{F}$ be a κ-regular w-set system $\mathcal{F}$ which does not contain $2r$ pairwise disjoint sets, where $\kappa > \max\left(\eta(w, r), 2\alpha(w, r)\right)$. We will show that this leads to a contradiction.

Let $\mathcal{D}$ be the corresponding κ-regular distribution on $\mathcal{F}$. Let $\mathcal{F}' \subseteq \mathcal{F}$ be any sub set-system with $\mathcal{D}(\mathcal{F}') \geq 1/2$. Claim 19 then implies that $\mathcal{F}'$ is $(\kappa/2)$-regular. By our choice of κ, $\kappa/2 > \alpha(w, r)$, and hence $\mathcal{F}'$ contains r pairwise disjoint sets.

More generally, consider the following setup. Let $\mathcal{D}' : \mathcal{F} \to \mathbb{R}_{\geq 0}$ with $\mathcal{D}'(S) \leq \mathcal{D}(S)$ for all $S \in \mathcal{F}$. Define $\mathcal{F}' = \{S : \mathcal{D}'(S) > 0\}$ and $\mathcal{D}'(\mathcal{F}) = \sum \mathcal{D}'(S)$. As long as $\mathcal{D}'(\mathcal{F}) \geq 1/2$ we are guaranteed that $\mathcal{F}'$ is $(\kappa/2)$-regular, and hence contains r pairwise disjoint sets. Consider the following process:

1. Initialize $D_0(S) = \mathcal{D}(S)$ for all $S \in \mathcal{F}$ and $i = 0$.
2. As long as $D_i(\mathcal{F}) \geq 1/2$ do:
   a. Let $\mathcal{F}_i = \{S : D_i(S) > 0\}$.
   b. Find r pairwise disjoint sets $S_{i,1}, \ldots, S_{i,r} \in \mathcal{F}_i$.
   c. Let $w_i = \min(D_i(S_{i,1}), \ldots, D_i(S_{i,r}))$.
   d. Set $D_{i+1}(S) = D_i(S) - w_i$ if $S \in \{S_{i,1}, \ldots, S_{i,r}\}$, and $D_{i+1}(S) = D_i(S)$ otherwise.
   e. Set $i \leftarrow i + 1$

Assume that the process terminates after m steps. Let $W_i = S_{i,1} \cup \ldots \cup S_{i,r}$, which by construction is a set of size wr. Note that as we assume that $\mathcal{F}$ does not contain $2r$ pairwise disjoint sets, we obtain that $W_1, \ldots, W_m$ must be an intersecting set system (possibly with some repeated sets). Let $w = \sum w_i$. As $D_{i+1}(\mathcal{F}) = D_i(\mathcal{F}) - w_i r$, and as we terminate when $D_m(\mathcal{F}) < 1/2$, we have

$$w \geq 1/2r.$$

Let $\mathcal{F}^* = \{W_1, \ldots, W_m\}$, namely taking each set exactly once. As it may be the case that $W_1, \ldots, W_m$ are not all distinct, we only know that $|\mathcal{F}^*| \leq m$. Consider the distribution D^* on $\mathcal{F}^*$ given by $D^*(W) = \frac{1}{w} \sum_{i:W_i=W} w_i$. Then as $\mathcal{F}^*$ is an intersecting set system, we obtain that D^* cannot be β-regular for $\beta = \beta(wr)$.

Thus, there exists a nonempty set T of size $|T| = t \geq 1$ such that

$$\sum_{W \in \mathcal{F}^*:\, T \subseteq W} D^*(W) \geq \beta^{-t}.$$

This implies that if we denote $I = \{i \in [m] :\, T \subseteq W_i\}$ then

$$\sum_{i \in I} w_i \geq w\beta^{-t} \geq \frac{1}{2r\beta^t}.$$

Next, consider some $i \in I$. Recall that W_i is the union of pairwise disjoint sets $S_{i,1}, \ldots, S_{i,r} \in \mathcal{F}$. In particular, there must exist $j_i \in [r]$ such that $|T \cap S_{i,j_i}| \geq |T|/r$. We denote $T_i = T \cap S_{i,j_i}$. As the number of possibles subsets of T is $2^{|T|}$, there must exist $T^* \subseteq T$ such that

$$\sum_{i \in I:\, T_i = T^*} w_i \geq 2^{-t} \sum_{i \in I} w_i \geq \frac{1}{2r(2\beta)^t}.$$

In particular, $|T^*| \geq |T|/r$ and

$$\sum_{i \in I:\, T^* \subseteq S_{i,j_i}} w_i \geq \frac{1}{2r(2\beta)^t}.$$

It may be that the list of S_{i,j_i} contains repeated sets (namely, that $S_{i,j_i} = S_{i',j_{i'}}$ for some $i \neq i'$). For each $S \in \mathcal{F}$ let $I(S) = \{i \in I : S_{i,j_i} = S\}$. In particular, $I(S)$ is not empty only for sets S with $T^* \subseteq S$. We can rewrite the sum as

$$\sum_{i \in I:\, T^* \subseteq S_{i,j_i}} w_i = \sum_{S \in \mathcal{F}:\, T^* \subseteq S} \sum_{i \in I(S)} w_i.$$

Next, fix some $S \in \mathcal{F}$ with $T^* \subseteq S$ and consider the internal sum. Recall that $w_i = D_i(S) - D_{i+1}(S)$, and hence the sum is a telescopic sum and can be bounded by

$$\sum_{i \in I(S)} w_i \leq D_0(S) - D_m(S) \leq D(S).$$

We thus obtain that

$$\sum_{S \in \mathcal{F}: T^* \subseteq S} \mathcal{D}(S) \geq \sum_{i \in I:\, T^* \subseteq S_{i,j_i}} w_i \geq \frac{1}{2r(2\beta)^t}.$$

Recall that $\mathcal{D}$ is κ-regular. We can upper bound κ by

$$\kappa \leq \left(2r(2\beta)^t\right)^{1/|T^*|} \leq \left(2r(2\beta)^t\right)^{r/t} \leq 2r(2\beta)^r = \eta(w,r).$$

Putting everything together, we get

$$\alpha(w, 2r) \leq \max\left(\eta(w,r), 2\alpha(w,r)\right).$$

To conclude the proof, note that if r is a power of two then by induction and our choice of η we have

$$\alpha(w, 2r) \leq \max\left(\eta(w,r), 2\eta(w,r/2), 4\eta(w,r/4), \ldots\right) = \eta(w,r).$$

Thus for a general r, if $r \leq s \leq 2r$ is the smallest power of two that upper bounds r then

$$\alpha(w,r) \leq \alpha(w,s) \leq \eta(w,s/2) \leq \eta(w,r).$$ ◀

Proof of Theorem 6. We put all the pieces together. Let $w \geq 2$. Assume Conjecture 21 holds. Lemma 22 gives that

$$\gamma(w) \leq (\log w)^c$$

for some constant $c \geq 1$. Claim 24 then gives that

$$\beta(w) \leq \gamma(w)$$

and Lemma 30 gives that

$$\alpha(w, r) \leq r2^{r+1}\beta(wr)^r \leq r2^{r+1}(\log(wr))^{cr} \leq (\log w)^{c_r}$$

for some constant $c_r \geq 1$. Finally, Claim 29 shows that if $\mathcal{F}$ is a w-set system of size $|\mathcal{F}| \geq (\log w)^{c_r w}$ then $\mathcal{F}$ contains an r-sunflower. ◀

7 Further discussions

Recall that $\beta(w)$ is the maximal κ such that there exists an intersecting κ-regular w-set system.

▶ **Conjecture 33.** $\beta(w) \leq (\log w)^{O(1)}$.

We would like to point out that in Conjecture 33, the assumption that the set system is intersecting cannot be replaced by a weaker assumption that it is almost intersecting, namely that most pairs of sets intersect. To see that, consider the following example.

▶ **Example 34.** Let $\mathcal{F}$ be the family of all sets of size w in a universe of size $n = cw^2$. By choosing an appropriate constant $c > 0$, we get that 99% of the sets $S, S' \in \mathcal{F}$ intersect. However, $\mathcal{F}$ is (w/c)-regular.

The following is an interesting family of examples, that might help shed light on Conjecture 33.

▶ **Example 35.** Let $\mathbb{F}_p$ be a finite field and $n \geq 1$. Let $V \subset \mathbb{F}_p^n$ be a linear subspace of dimension k. Given a set of coordinates $I \subseteq [n]$, define $V_I = \{(v_i)_{i \in I} : v \in V\}$ to be the subspace obtained by restricting vectors $v \in V$ to coordinates I. We say that V is *α-large* if

$$\dim(V_I) \geq \alpha|I| \qquad \forall I \subseteq [n].$$

In particular, this implies that $k \geq \alpha n$.

Next, we define a set system corresponding to a subspace. Let $X = \{(i, a) : i \in [n], a \in \mathbb{F}_p\}$. For any vector $v \in \mathbb{F}_p^n$ define its corresponding set

$$S(v) = \{(i, v_i) : i \in [n]\} \subset X.$$

For a subspace $V \subset \mathbb{F}_p^n$ define the set system

$$\mathcal{F}(V) = \{S(v) : v \in V\}.$$

Observe that:

- **(i)** $\mathcal{F}(V)$ is an n-set system of size p^k.
- **(ii)** For any $T \subseteq X$ it holds that $|\{S \in \mathcal{F}(V) : T \subseteq S\}| \leq p^{-\alpha|T|}|\mathcal{F}|$. Hence $\mathcal{F}(V)$ is κ-regular for $\kappa = p^\alpha$.
- **(iii)** $\mathcal{F}(V)$ is intersecting iff any $v \in V$ contains at least one zero coordinate.

If Conjecture 33 holds and $p \geq (\log n)^c$ for some absolute constant $c > 0$, then it must hold that V contains a vector with no zero coordinates. This motivates the following problem.

▶ **Problem 36.** *Let $V \subset \mathbb{F}_p^n$ be a α-large subspace. Prove that if $p \geq (\log n)^c$, for some $c = c(\alpha)$, then V must contain a vector with no zero coordinates.*

A previous version of this paper gave a more restricted version of Example 35, corresponding to the case when V spans an MDS code. Namely, $\dim(V_I) = |I|$ for all $I \subseteq [n]$ with $|I| \leq k$. Ryan Alweiss [2] proved the analog of Problem 36 for this case, in fact where $p \geq p_0(n/k)$.

References

1 Noga Alon, Amir Shpilka, and Christopher Umans. On sunflowers and matrix multiplication. *computational complexity*, 22(2):219–243, 2013.

2 Ryan Alweiss. Personal communication, 2019.

3 Paul Erdős and Richard Rado. Intersection theorems for systems of sets. *Journal of the London Mathematical Society*, 35(1):85–90, 1960.

4 Mika Goos, Shachar Lovett, Raghu Meka, Thomas Watson, and David Zuckerman. Rectangles are nonnegative juntas. *SIAM Journal on Computing*, 45(5):1835–1869, 2016.

5 Parikshit Gopalan, Raghu Meka, and Omer Reingold. DNF sparsification and a faster deterministic counting algorithm. *Computational Complexity*, 22(2):275–310, 2013.

6 Alexandr V Kostochka. Extremal Problems on Δ-Systems. In *Numbers, Information and Complexity*, pages 143–150. Springer, 2000.

7 Xin Li, Shachar Lovett, and Jiapeng Zhang. Sunflowers and Quasi-Sunflowers from Randomness Extractors. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

8 Shachar Lovett and Jiapeng Zhang. DNF sparsification beyond sunflowers. In *Electronic Colloquium on Computational Complexity (ECCC)*, 2018.

9 Alexander A Razborov. Lower bounds for the monotone complexity of some Boolean functions. In *Soviet Math. Dokl.*, volume 31, pages 354–357, 1985.

10 Benjamin Rossman. The monotone complexity of k-clique on random graphs. In *Proceedings of the 2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 193–201. IEEE Computer Society, 2010.

Resolution and the Binary Encoding of Combinatorial Principles

Stefan Dantchev
Department of Computer Science University of Durham, UK
s.s.dantchev@durham.ac.uk

Nicola Galesi
Dipartimento di Informatica, Sapienza Università di Roma, Italy
nicola.galesi@uniroma1.it

Barnaby Martin
Department of Computer Science University of Durham, UK
barnaby.d.martin@durham.ac.uk

Abstract

$\mathsf{Res}(s)$ is an extension of Resolution working on s-DNFs. We prove tight $n^{\Omega(k)}$ lower bounds for the size of refutations of the binary version of the k-Clique Principle in $\mathsf{Res}(o(\log\log n))$. Our result improves that of Lauria, Pudlák et al. [27] who proved the lower bound for $\mathsf{Res}(1)$, i.e. Resolution. The exact complexity of the (unary) k-Clique Principle in Resolution is unknown. To prove the lower bound we do not use any form of the Switching Lemma [35], instead we apply a recursive argument specific for binary encodings. Since for the k-Clique and other principles lower bounds in Resolution for the unary version follow from lower bounds in $\mathsf{Res}(\log n)$ for their binary version we start a systematic study of the complexity of proofs in Resolution-based systems for families of contradictions given in the binary encoding.

We go on to consider the binary version of the weak Pigeonhole Principle $\mathsf{Bin\text{-}PHP}_n^m$ for $m > n$. Using the the same recursive approach we prove the new result that for any $\delta > 0$, $\mathsf{Bin\text{-}PHP}_n^m$ requires proofs of size $2^{n^{1-\delta}}$ in $\mathsf{Res}(s)$ for $s = o(\log^{1/2} n)$. Our lower bound is almost optimal since for $m \geq 2^{\sqrt{n \log n}}$ there are quasipolynomial size proofs of $\mathsf{Bin\text{-}PHP}_n^m$ in $\mathsf{Res}(\log n)$.

Finally we propose a general theory in which to compare the complexity of refuting the binary and unary versions of large classes of combinatorial principles, namely those expressible as first order formulae in Π_2-form and with no finite model.

2012 ACM Subject Classification Theory of computation → Complexity theory and logic

Keywords and phrases Proof complexity, k-DNF resolution, binary encodings, Clique and Pigeonhole principle

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.6

Acknowledgements We are grateful to Ilario Bonacina for reading a preliminary version of this work and addressing us some useful comments and observations. We are further grateful to several anonymous reviewers for further corrections and comments.

1 Introduction

Various fundamental combinatorial principles used in Proof Complexity may be given in first-order logic as sentences φ with no finite models. Riis discusses in [34] how to generate from φ a family of CNFs, the nth of which encodes that φ has a model of size n, which are hence contradictions. Following Riis, it is typical to encode the existence of the witnesses in longhand with a big disjunction, that we designate the *unary encoding*. As recently investigated in the works [19, 12, 13, 27, 22], it may also be possible to encode the existence of such witnesses *succinctly* by the use of a *binary encoding*. Essentially, the existence of the witness is now given implicitly as any propositional assignment to the relevant variables

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka; Article No. 6; pp. 6:1–6:25


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

gives a witness, whereas in the unary encoding a solitary true literal tells us which is the witness. Combinatorial principles encoded in binary are interesting to study since, loosely speaking, they still preserve the hardness of the combinatorial principle encoded while giving a more succinct propositional representation. In certain cases this leads to obtain significant lower bounds in an easier way than for the unary case [19, 13, 27].

The central thrust of this work is to contrast the proof complexity (size) between the unary and binary encodings of natural combinatorial principles. This has not previously been done systematically in Proof Complexity, though it has been better-studied the in the "dual" area of SAT-solving [26, 29]. In the SAT community it is well-known one may try various different encodings of the 1-from-n constraint to speed-up proofs of unsatisfiability as well as satisfiability. In [29, 37], what we call the binary encoding is referred to as *logarithmic*. The Pigeonhole Principle is explored experimentally in both [26] and Chapter 7 in [29, 37] (though sadly the binary encoding is not among the tests).

A principal motivation is to approach size lower bounds of refutations in Resolution for families of contradictions in the usual unary encoding, by looking at the complexity of proofs in $\mathsf{Res}(s)$ for the corresponding families of contradictions where witnesses are given in the binary encodings. $\mathsf{Res}(s)$, is a refutational proof system extending Resolution to s-bounded DNFs, introduced by Krajíček in [23]. Our approach is justified by observing that (see Lemma 26), for a family of contradictions encoding a principle which is expressible as Π_2 first-order formulae having no finite models, short $\mathsf{Res}(\log n)$ refutations of their *binary* encoding can be obtained from short Resolution refutations for the *unary* encoding. Lower bounds for $\mathsf{Res}(s)$ have appeared variously in the literature. Of most interest to us are those for the (moderately weak) Pigeonhole Principle PHP^{2n}_n, for $\mathsf{Res}(\sqrt{\log n/\log\log n})$ in [35], improved to $\mathsf{Res}(\epsilon \log n/\log\log n)$ in [2]. A hierarchy in $\mathsf{Res}(s)$ is uncovered by the use of relativising the (Linear) Ordering Principle in [17].

Our first interest is the *k-Clique Principle*, whose precise Resolution complexity is still unknown; but we also study other principles, to make progress in the direction of our approach. The three combinatorial principles we deal with in this paper are: (1) the k-Clique Formulae, $\mathsf{Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$; (2) the (weak) Pigeonhole Principle PHP^m_n; and (3) the (Linear) Ordering Principle, $(\mathsf{L})\mathsf{OP}_n$. The *$k$-Clique Formulae* introduced in [10, 11, 6] are formulae stating that a given graph G does have a k-clique and are therefore unsatisfiable when G does not contain a k-clique. The Pigeonhole Principle states that a total mapping $f : [m] \to [n]$ has necessarily a collision when $m > n$. Its propositional formulation in the negation, PHP^m_n is well-studied in proof complexity (see among others: [21, 35, 16, 31, 33, 32, 8, 15, 9, 7, 5, 3, 28]). The $\mathsf{LOP_n}$ formulae encode the negation of the Linear Ordering Principle which asserts that each finite linearly ordered set has a maximal element and was introduced and studied, among others, in the works [24, 36, 14].

1.1 *k*-Clique Principle

Deciding whether a graph has a k-clique it is one of the central problems in Computer Science and can be decided in time $n^{O(k)}$ by a brute force algorithm. It is then of the utmost importance to understand whether given algorithmic primitives are sufficient to design algorithms solving the Clique problem more efficiently than the trivial upper bound. Resolution refutations for the formula $\mathsf{Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$ (respectively any CNF F), can be thought as the execution trace of an algorithm, whose primitives are defined by the rules of the Resolution system, searching for a k-Clique inside G (respectively deciding the satisfiability of F). Hence understanding whether there are $n^{\Omega(k)}$ size lower bounds in Resolution for refuting $\mathsf{Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$ would then answer the above question for algorithms based on Resolution primitives. This

question was posed in [10], where it was also answered in the case of refutations in the form of trees (treelike Resolution). Recently in a major breakthrough Atserias et al. in [4] prove the $n^{\Omega(k)}$ lower bound for the case of read-once proofs (Regular resolution). The graph G considered in [10, 4] to plug in the formula $\mathsf{Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$ to make it unsatisfiable was a random graph obtained by a slight variation of Erdös-Rényi distribution of random graphs as defined in [10]. But the exact Resolution complexity of $\mathsf{Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$, for G random is unknown. In the work [27], Lauria et al. consider the binary encoding of Ramsey-type propositional statements, having as a special case a binary version of $\mathsf{Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$: $\mathsf{Bin\text{-}Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$. They obtain optimal lower bounds for $\mathsf{Bin\text{-}Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$ in Resolution, which is $\mathsf{Res}(1)$.

Our main result (Theorem 7) is a $n^{\Omega(k)}$ lower bound for the size of refutations of $\mathsf{Bin\text{-}Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$ in $\mathsf{Res}(o(\log\log n))$, when G is a random graph as that defined in [10]. Lemma 2 in Section 3 proves that a lower bound in $\mathsf{Res}(\log)$ for the $\mathsf{Bin\text{-}Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$ would prove a lower bound in Resolution for $\mathsf{Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$.

1.2 Weak Pigeonhole Principle

An interesting example to test the relative hardness of binary versions of combinatorial principle comes from the (weak) Pigeonhole Principle. In Section 4, we consider its binary version $\mathsf{Bin\text{-}PHP}^m_n$ and we prove that in $\mathsf{Res}(s)$, for all $\epsilon > 0$ and $s \leq \log^{\frac{1}{2-\epsilon}}(n)$, the shortest proofs of the $\mathsf{Bin\text{-}PHP}^m_n$, require size $2^{n^{1-\delta}}$, for any $\delta > 0$ (Theorem 22). This is the first size lower bound known for the $\mathsf{Bin\text{-}PHP}^m_n$ in $\mathsf{Res}(s)$. As a by-product of this lower bound we prove a lower bound of the order $2^{\Omega(\frac{n}{\log n})}$ (Theorem 18) for the size of the shortest Resolution refutation of $\mathsf{Bin\text{-}PHP}^m_n$. Our lower bound for $\mathsf{Res}(s)$ is obtained through a technique that merges together the random restriction method, an inductive argument on the s of $\mathsf{Res}(s)$ and the notion of *minimal covering* of a k-DNF of [35]. Since we are not using any (even weak) form of Switching Lemma (as for instance in [35, 1]), we consider how tight is our lower bound in $\mathsf{Res}(s)$. We prove that $\mathsf{Bin\text{-}PHP}^m_n$ (Theorem 23) can be refuted in size $2^{O(n)}$ in treelike $\mathsf{Res}(1)$. Our upper bound is contrasting with the unary case of the Pigeonhole Principle, PHP^m_n, which instead requires treelike $\mathsf{Res}(1)$ refutations of size $2^{\Omega(n\log n)}$, as proved in [9, 16].

For the Pigeonhole Principle, similarly to the k-Clique Principle, we can prove that short $\mathsf{Res}(\log n)$ refutations for $\mathsf{Bin\text{-}PHP}^m_n$ can be efficiently obtained from short $\mathsf{Res}(1)$ of PHP^m_n (Lemma 15). This allows us to prove that our lower bound is almost optimal: Buss and Pitassi, in [15], proved an upper bound of $2^{O(\sqrt{n\log n})}$ for the size of refuting PHP^m_n in $\mathsf{Res}(1)$ when $m \geq 2^{\sqrt{n\log n}}$, which by our Lemma 15 holds also for $\mathsf{Res}(\log n)$ proofs of $\mathsf{Bin\text{-}PHP}^m_n$. It follows that our exponential lower bound for $\mathsf{Bin\text{-}PHP}^m_n$ (Theorem 18) for any $m > n$ in $\mathsf{Res}(\log^{1/2-\epsilon} n)$ is almost optimal.

1.3 Contrasting unary and binary principles

To work with a more general theory in which to contrast the complexity of refuting the binary and unary versions of combinatorial principles, following Riis [34] we consider principles which are expressible as first-order formulae with no finite model in Π_2-form, i.e. as $\forall\vec{x}\exists\vec{w}\varphi(\vec{x},\vec{w})$ where $\varphi(\vec{x},\vec{y})$ is a formula built on a family of relations $\vec{R}$. For example the *Ordering Principle*, which states that a finite partial order has a maximal element is one such principle. Its negation can be expressed in Π_2-form as: $\forall x,y,z\exists w\ \neg R(x,x) \wedge (R(x,y) \wedge R(y,z) \rightarrow R(x,z)) \wedge R(x,w)$. In Definition 25 we explain how to generate a binary encoding $\mathsf{Bin\text{-}C_n}$ from any combinatorial principle $\mathsf{C_n}$ expressible as a first order formulae in Π_2-form with no finite models and whose unary encoding we denote by $\mathsf{Un\text{-}C_n}$. Another example is the

Pigeonhole Principle whose negation of its relational form can be expressed as a Π_2-formula as $\forall x, y, z \exists w \neg R(x, 0) \wedge (R(x, z) \wedge R(y, z) \rightarrow x = y) \wedge R(x, w)$. Notice that in the case of the Pigeonhole Principle, the existential witness w to the type *pigeon* is of the distinct type *hole*. Furthermore, pigeons only appear on the left-hand side of atoms $R(x, z)$ and holes only appear on the right-hand side. This accounts for why, in the case of the Pigeonhole Principle, one can give another more efficient (in terms of number of variables) binary encoding (see Section 5 for details), than the one given by $\mathsf{Bin\text{-}C_n}$ applied to the PHP. Nevertheless in Lemma 27 we observe that in Resolution efficient refutations for one encoding can be obtained from refutations of the other encoding. We propose a framework to compare lower bounds for the $\mathsf{Bin\text{-}C_n}$ in $\mathsf{Res}(s)$ with lower bounds for $\mathsf{Un\text{-}C_n}$ in $\mathsf{Res}(1)$ by proving in Lemma 26 that short Resolution refutations for $\mathsf{Un\text{-}C_n}$ produces short $\mathsf{Res}(\log n)$ refutations for $\mathsf{Bin\text{-}C_n}$.

1.3.1 Linear Ordering Principles

Linear ordering formulae $\mathsf{LOP_n}$ encodes a Linear Ordering Principle. They were used in [14, 20] as families of formulae witnessing the optimality of the size-width tradeoffs for Resolution ([8]), so that they require high width to be refuted, but still admit polynomial size refutations in Resolution. Here we face the following open question: is the binary encoding of $\mathsf{LOP_n}$ formula still efficiently refutable in Resolution? In answering this question we will show something stronger, as we study under what conditions the complexity of proofs in Resolution will not increase significantly (by more than a polynomial factor) when shifting from the unary encoding to the binary encoding. In Lemma 24 we prove that this is true for the negation of principles expressible as first order formula in Π_2-form involving *total variable comparisons*. Hence in particular the binary version of the Linear Ordering Principle $\mathsf{Bin\text{-}LOP_n}$. Finally, we also prove that the binary encoding of the Ordering Principle $\mathsf{Bin\text{-}OP_n}$, where antisymmetry is not encoded and hence there is no total variable comparison, is also polynomially provable in Resolution. Broadly speaking, these results are saying that shifting to the binary encodings is not destroying the hardness of a unary principle when working in Resolution. Hence binary encodings of combinatorial principles are meaningful benchmarks for Resolution to prove lower bounds for.

1.3.2 Binary encodings of principles versus their Unary functional encodings

The *unary functional* encoding of a combinatorial principle replaces the big disjunctive clauses of the form $v_{i,1} \vee \ldots \vee v_{i,n}$, with $v_{i,1} + \ldots + v_{i,n} = 1$, where addition is made on the natural numbers. This is equivalent to augmenting the axioms $\neg v_{i,j} \vee \neg v_{i,k}$, for $j \neq k \in [n]$. One might argue that the unary functional encoding is the true unary analog to the binary encoding, since the binary encoding naturally enforces that there is a single witness alone. It is likely that the non-functional formulation was preferred for its simplicity (similarly as the Pigeonhole Principle is often given in its non-functional formulation).

In Subsection 5.3, we prove that the Resolution refutation size increases by only a quadratic factor when moving from the binary encoding to the unary functional encoding. This is interesting because the same does not happen for treelike Resolution, where the unary encoding has complexity $2^{\Theta(n \log n)}$ [9, 16], while, as we prove in Subsection 4.1 (Theorem 23), the binary (functional) encoding is $2^{\Theta(n)}$. The unary encoding complexity is noted in [17] and remains true for the unary functional encoding with the same lower-bound proof. The binary encoding complexity is addressed directly in this paper.

1.4 Techniques and Organization

The method of random restrictions in Proof Complexity is often employed to prove size lower bounds. Loosely speaking the method works as follows: we consider formulae having a given specific combinatorial property P; after hitting, with a suitable random partial assignment, on an allegedly short proof of the formula we are refuting, we are left to prove that with high probability a formula with property P is killed away from the proof. The growth rate as the probability approaches to 1 together with a counting argument using averaging (as the union bound), implies a lower bound on the number of formulae with property P in the proof. Lower bounds in $\mathsf{Res}(s)$ using random restrictions were known only for $s = 2$ (see [5]). Using a weak form of the Switching Lemma, lower bounds for $\mathsf{Res}(s)$ were obtained in [35, 1]. From the latter paper we use the notion of *covering number* of a k-DNF F, i.e. the minimal size of a set of variables to hit all the k-terms in F. In this work we merge the covering number with the random restriction method together with an inductive argument on the s, to get size lower bounds in $\mathsf{Res}(s)$ specifically for binary encoding of combinatorial principles.

After a section with the preliminaries, the paper is divided into four sections: one with the lower bound for the k-Clique Principle, one containing all the results for the (weak) Pigeonhole Principle, one for the contrasting the proof complexity between unary and binary principles containing all the results about the various Ordering Principles, and finally the last section containing a general approach to unary vs binary encodings for principle expressible as a Π_2-formulae.

2 Preliminaries

We denote by $\top$ and $\bot$ the Boolean values "true" and "false", respectively. A *literal* is either a propositional variable or a negated variable. We will denote literals by small letters, usually l's. An *s-conjunction* (*s-disjunction*) is a conjunction (disjunction) of at most k literals. A *clause* with s literals is a s-disjunction. The width $w(C)$ of a clause C is the number of literals in C. A *term* (*s-term*) is either a conjunction (s-conjunction) or a constant, $\top$ or $\bot$. A *s-DNF* or *s-clause* (*s-CNF*) is a disjunction (conjunction) of an unbounded number of s-conjunctions (s-disjunctions). We will use calligraphic capital letters to denote s-CNFs or s-DNFs, usually $\mathcal{C}$s for CNFs, $\mathcal{D}$s for DNFs and $\mathcal{F}$s for both. For example, $((v_1 \wedge \neg v_2) \vee (v_2 \wedge v_3) \vee (\neg v_1 \wedge v_3))$ is an example of a 2-DNF and its negation $((v_1 \vee \neg v_2) \wedge (v_2 \vee v_3) \wedge (\neg v_1 \vee v_3))$ is an example of a 2-CNF.

We can now describe the propositional refutation system $\mathsf{Res}(s)$ ([23]). It is used *to refute* (i.e. to prove inconsistency) of a given set of s-clauses by deriving the empty clause from the initial clauses. There are four derivation rules:

1. The *$\wedge$-introduction rule* is
$$\frac{\mathcal{D}_1 \vee \bigwedge_{j\in J_1} l_j \quad \mathcal{D}_2 \vee \bigwedge_{j\in J_2} l_j}{\mathcal{D}_1 \vee \mathcal{D}_2 \vee \bigwedge_{j\in J_1\cup J_2} l_j},$$
provided that $|J_1 \cup J_2| \leq s$.
2. The *cut (or resolution) rule* is
$$\frac{\mathcal{D}_1 \vee \bigvee_{j\in J} l_j \quad \mathcal{D}_2 \vee \bigwedge_{j\in J} \neg l_j}{\mathcal{D}_1 \vee \mathcal{D}_2},$$
3. The two *weakening rules* are
$$\frac{\mathcal{D}}{\mathcal{D} \vee \bigwedge_{j\in J} l_j} \quad \text{and} \quad \frac{\mathcal{D} \vee \bigwedge_{j\in J_1\cup J_2} l_j}{\mathcal{D} \vee \bigwedge_{j\in J_1} l_j},$$
provided that $|J| \leq s$.

A $\mathsf{Res}(s)$ refutation can be considered as a directed acyclic graph (DAG), whose sources are the initial clauses, called also axioms, and whose only sink is the empty clause. We shall define *the size of a proof* to be the number of the internal nodes of the graph, i.e. the number of applications of a derivation rule, thus ignoring the size of the individual s-clauses in the refutation. In principle the s from "$\mathsf{Res}(s)$" could depend on n – an important special case is $\mathsf{Res}(\log n)$.

Clearly, $\mathsf{Res}(1)$ is *(ordinary) Resolution*, working on clauses, and using only the cut rule, which becomes the usual resolution rule, and the first weakening rule. Given an unsatisfiable CNF $\mathcal{C}$, and a $\mathsf{Res}(1)$ refutation π of $\mathcal{C}$ the width of π, $w(\pi)$ is the maximal width of a clause in π. The width refuting $\mathcal{C}$ in Res(1), $w(\vdash \mathcal{C})$, is the minimal width over all $\mathsf{Res}(1)$ refutations of $\mathcal{C}$.

A *covering set* for a s-DNF $\mathcal{D}$ is a set of literals L such that each term of $\mathcal{D}$ has at least a literal in L. The *covering number* $c(\mathcal{D})$ of a s-DNF $\mathcal{D}$ is the minimal size of a covering set for $\mathcal{D}$.

Let $\mathcal{F}(x_1 \dots, x_n)$ be a boolean s-DNF (resp. s-CNF) defined over variables $X = \{x_1, \dots, x_n\}$. A *partial assignment* ρ to $\mathcal{F}$ is a truth-value assignment to some of the variables of $\mathcal{F}$: $dom(\rho) \subseteq X$. By $\mathcal{F}\restriction_\rho$ we denote the formula $\mathcal{F}'$ over variables in $X \setminus dom(\rho)$ obtained from $\mathcal{F}$ after simplifying in it the variables in $dom(\rho)$ according to the usual boolean simplification rules of clauses and terms.

2.1 $\mathsf{Res}(s)$ vs Resolution

Similarly to what was done for treelike $\mathsf{Res}(s)$ refutations in [18], if we turn a Res (s) refutation of a given set of s-clauses Σ upside-down, i.e. reverse the edges of the underlying graph and negate the s-clauses on the vertices, we get a special kind of restricted branching s-program whose nodes are labelled by s-CNFs and at each node some s-disjunction is questioned. The restrictions are as follows.

Each vertex is labelled by a s-CNF which partially represents the information that can be obtained along any path from the source to the vertex (this is a *record* in the parlance of [30]). Obviously, the (only) source is labelled with the constant $\top$. There are two kinds of queries, which can be made by a vertex:

1. Querying a new s-disjunction, and branching on the answer, which can be depicted as follows.

$$\begin{array}{ccc} & \mathcal{C} & \\ & ?\bigvee_{j\in J} l_j & \\ \top\swarrow & & \searrow\bot \\ \mathcal{C}\wedge\bigvee_{j\in J} l_j & & \mathcal{C}\wedge\bigwedge_{j\in J}\neg l_j \end{array} \tag{1}$$

2. Querying a known s-disjunction, and splitting it according to the answer:

$$\begin{array}{ccc} & \mathcal{C}\wedge\bigvee_{j\in J_1\cup J_2} l_j & \\ & ?\bigvee_{j\in J_1} l_j & \\ \top\swarrow & & \searrow\bot \\ \mathcal{C}\wedge\bigvee_{j\in J_1} l_j & & \mathcal{C}\wedge\bigvee_{j\in J_2} l_j \end{array} \tag{2}$$

There are two ways of forgetting information,

$$\begin{array}{c} \mathcal{C}_1\wedge\mathcal{C}_2 \\ \downarrow \\ \mathcal{C}_1 \end{array} \quad \text{and} \quad \begin{array}{c} \mathcal{C}\wedge\bigvee_{j\in J_1} l_j \\ \downarrow \\ \mathcal{C}\wedge\bigvee_{j\in J_1\cup J_2} l_j \end{array}, \tag{3}$$

the point being that forgetting allows us to equate the information obtained along two different branches and thus to merge them into a single new vertex. A sink of the branching s-program must be labelled with the negation of a s-clause from Σ. Thus the branching s-program is supposed by default to solve the *Search problem for* Σ: given an assignment of the variables, find a clause which is falsified under this assignment.

The equivalence between a Res (s) refutation of Σ and a branching s-program of the kind above is obvious. Naturally, if we allow querying single variables only, we get branching 1-programs – decision DAGs – that correspond to Resolution. If we do not allow the forgetting of information, we will not be able to merge distinct branches, so what we get is a class of decision trees that correspond precisely to the treelike version of these refutation systems.

Finally, we mention that the queries of the form (1) and (2) as well as forget-rules of the form (3) give rise to a Prover-Adversary game (see [30] where this game was introduced for Resolution). In short, Adversary claims that Σ is satisfiable, and Prover tries to expose him. Prover always wins if her strategy is kept as a branching program of the form we have just explained, whilst a good (randomised) Adversary's strategy would show a lower bound on the branching program, and thus on any Res (k) refutation of Σ.

▶ **Lemma 1.** *If a CNF ϕ has a refutation in* Res$(k+1)$ *of size N, whose corresponding branching $(k+1)$-program has no records of covering number $\geq d$, then ϕ has a* Res(k) *refutation of size $2^{d+2} \cdot N$ (which is $\leq e^d$ when $d > 4$).*

Proof. In the branching program, consider a $(k+1)$-CNF record ϕ whose covering number $< d$ is witnessed by variable set $V' := \{v_1, \ldots, v_d\}$. At this node some $(k+1)$-disjunction $(l_1 \vee \ldots \vee l_k \vee l_{k+1})$ is questioned.

Now in place of the record ϕ in our original branching program we expand a mini-tree of size 2^{d+2} with 2^{d+1} leaves questioning all the variables of V' as well as the literal l_{k+1}. Clearly, each evaluation of these reduces ϕ to a k-CNF that logically implies ϕ. It remains to explain how to link the leaves of these mini-trees to the roots of other mini-trees. At each leaf we look to see whether we have the information l_{k+1} or $\neg l_{k+1}$. If l_{k+1} then we link immediately to the root of the mini-tree corresponding to the yes-answer to $(l_1 \vee \ldots \vee l_k \vee l_{k+1})$ (without asking a question). If $\neg l_{k+1}$ then we question $(l_1 \vee \ldots \vee l_k$ and, if this is answered yes, link the yes-answer to $(l_1 \vee \ldots \vee l_k \vee l_{k+1})$, otherwise to its no-answer. ◀

3 The binary encoding of k-Clique

Consider a graph G such that G is formed from k blocks of n nodes each: $G = (\bigcup_{b\in[k]} V_b, E)$, where edges may only appear between distinct blocks. Thus, G is a k-partite graph. Let the edges in E be denoted as pairs of the form $E((i,a),(j,b))$, where $i \neq j \in [k]$ and $a, b \in [n]$.

The (unary) k-Clique CNF formulae $\mathsf{Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$ for G, has variables $v_{i,q}$ with $i \in [k], a \in [n]$, with clauses $\neg v_{i,a} \vee \neg v_{j,b}$ whenever $\neg E((i,a),(j,b))$ (i.e. there is no edge between node a in block i and node b in block j), and clauses $\bigvee_{a\in[n]} v_{i,a}$, for each block i. This expresses that $\mathcal{G}^n_k$ has a k-clique (with one vertex in each block), which we take to be a contradiction, since we will arrange for G not to have a k-clique.

$\mathsf{Bin\text{-}Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$ variables $\omega_{i,j}$ range over $i \in [k], j \in [\log n]$. Let us assume for simplicity of our exposition that n is a power of 2, the general case is explained in Section 5.2. Let $a \in [n]$ and let $a_1 \ldots a_{\log n}$ be its binary representation. Each (unary) variable $v_{i,j}$ semantically corresponds to the conjunction $(\omega_{i,1}^{a_1} \wedge \ldots \wedge \omega_{i,\log n}^{a_{\log n}})$, where

$$\omega_{i,j}^{a_j} = \begin{cases} \omega_{i,j} & \text{if } a_j = 1 \\ \overline{\omega}_{i,j} & \text{if } a_j = 0 \end{cases}$$

Hence in $\mathsf{Bin\text{-}Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$ we encode the unary clauses $\neg v_{i,a} \vee \neg v_{j,b}$, by the clauses

$$(\omega_{i,1}^{1-a_1} \vee \ldots \vee \omega_{i,\log n}^{1-a_{\log n}}) \vee (\omega_{j,1}^{1-b_1} \vee \ldots \vee \omega_{j,\log n}^{1-b_{\log n}})$$

The wide clauses from the unary encoding simply disappear in the binary encoding being implicit.

By the next Lemma short Resolution refutations for $\mathsf{Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$ can be translated into short $\mathsf{Res}(\log n)$ refutations of $\mathsf{Bin\text{-}Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$. hence to obtain lower bounds for $\mathsf{Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$ in Resolution, it suffices to obtain lower bounds for $\mathsf{Bin\text{-}Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$ in $\mathsf{Res}(\log n)$.

▶ **Lemma 2.** *Suppose there are Resolution refutations of* $\mathsf{Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$ *of size* S*. Then there are* $\mathsf{Res}(\log n)$ *refutations of* $\mathsf{Bin\text{-}Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$ *of size* S*.*

Proof. Where the decision DAG for $\mathsf{Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$ questions some variable $v_{i,a}$, the decision branching $\log n$-program questions instead $(\omega_{1,1}^{1-a_1} \vee \ldots \vee \omega_{1,\log n}^{1-a_{\log n}})$ where the out-edge marked true in the former becomes false in the latter, and vice versa. What results is indeed a decision branching $\log n$-program for $\mathsf{Bin\text{-}Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$, and the result follows. ◀

Following [10, 4, 27] we consider $\mathsf{Bin\text{-}Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$ formulae where G is a random graph distributed according to a variation of the Erdös-Rényi as defined in [10]. In the standard model, random graphs on n vertices are constructed by including every edge independently with probability p. It is known that k-cliques appear at the threshold probability $p^* = n^{-\frac{2}{k-1}}$. If $p < p^*$, then with high probability there is no k-clique. By $\mathcal{G}^n_{k,\epsilon}(p)$ we denote the distribution on random multipartite Erdős-Renyi graph with k blocks V_i of n vertices each, where each edge is present with probability p depending on ϵ. For $p = n^{-(1+\epsilon)\frac{2}{k-1}}$ we just write $\mathcal{G}^n_{k,\epsilon}$.

We use the notation $G = (\bigcup_{b\in[k]} V_b, E) \sim \mathcal{G}^n_k(p)$ to say that G is a graph drawn at random from the distribution $\mathcal{G}^n_k(p)$.

In the next sections we explore lower bounds for $\mathsf{Bin\text{-}Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$ in $\mathrm{Res}(s)$ for $s \geq 1$, when $G \sim \mathcal{G}^n_k(p)$.

3.1 Isolating the properties of G

Let α be a constant such that $0 < \alpha < 1$. Define a set of vertices U in G, $U \subseteq V$ to be an *α-transversal* if: (1) $|U| \leq \alpha k$, and (2) for all $b \in [k]$, $|V_b \cap U| \leq 1$. Let $B(U) \subseteq [k]$ be the set of blocks mentioned in U, and let $\overline{B(U)} = [k] \setminus B(U)$. We say that U is *extendible* in a block $b \in \overline{B(U)}$ if there exists a vertex $a \in V_b$ which is a common neighbour of all nodes in U, i.e. $a \in N_c(U)$ where $N_c(U)$ is the set of *common neighbours* of vertices in U i.e. $N_c(U) = \{v \in V \mid v \in \bigcap_{u\in U} N(u)\}$.

Let σ be a partial assignment (a restriction) to the variables of $\mathsf{Bin\text{-}Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$ and β a constant such that $0 < \beta < 1$. We call σ, *β-total* if σ assigns $\lfloor \beta \log n \rfloor$ bits in each block $b \in [k]$, i.e. $\lfloor \beta \log n \rfloor$ variables $\nu_{b,i}$ in each block b. Let $v = (i, a)$ be the a-th node in the i-the block in G. We say that a restriction σ is *consistent* with v if for all $j \in [\log n]$, $\sigma(\omega_{i,j})$ is either a_j or not assigned.

▶ **Definition 3.** *Let* $0 < \alpha, \beta < 1$*. A* α*-transversal set of vertices* U *is* β*-extendible, if for all* β*-total restriction* σ*, there is a node* v^b *in each block* $b \in \overline{B(U)}$*, such that* σ *is consistent with* v^b*.*

▶ **Lemma 4** (Extension Lemma). *Let $0 < \epsilon < 1$, let $k \leq \log n$. Let $1 > \alpha > 0$ and $1 > \beta > 0$ such that $1 - \beta > \alpha(2 + \epsilon)$. Let $G \sim \mathcal{G}^n_{k,\epsilon}$. With high probability both the following properties hold:*

1. *all α-transversal sets U are β-extendible;*
2. *$\mathcal{G}$ does not have a k-clique.*

Proof. Let U be an α-transversal set and σ be a β-total restriction. The probability that a vertex w is in $N_c(U)$ is $p^{\alpha k}$. Hence $w \notin N_c(U)$ with probability $(1 - p^{\alpha k})$. After σ is applied, in each block $b \in \overline{B(U)}$ remain $2^{\log n - \beta \log n} = n^{1-\beta}$ available vertices. Hence the probability that we cannot extend U in each block of $\overline{B(U)}$ after σ is applied is $(1 - p^{\alpha k})^{n^{1-\beta}}$. Fix $c = 2 + \epsilon$ and $\delta = 1 - \beta - \alpha c$. Notice that $\delta > 0$ by our choice of α and β. Since $p = \frac{1}{n^{\frac{c}{k}}}$, previous probability is $(1 - 1/n^{\alpha c})^{n^{1-\beta}}$, which is asymptotically $e^{-\frac{n^{1-\beta}}{n^{\alpha c}}} = e^{-n^{\delta}}$.

There are $\binom{k}{\alpha k}$ possible α-transversal sets U and $\binom{\log n}{\beta \log n} \cdot k$ possible β-total restrictions σ.

$$\begin{aligned}\binom{k}{\alpha k} \cdot \binom{\log n}{\beta \log n} \cdot k &\leq k^{\alpha k} \cdot (\log n)^{\beta \log n} \cdot k \\ &= 2^{\alpha k \log k + \beta \log n \log \log n + \log k} \\ &\leq 2^{\log^2 n}\end{aligned}$$

Notice that the last inequality holds since $k \leq \log n$. Hence the probability that there is in G no α-transeversal set U which is β-extendible is going to 0 as n grows.

To bound the probability that $\mathcal{G}$ contains a k-clique, notice that the expected number of k cliques is $\binom{n}{k} \cdot p^{\binom{k}{2}} \leq n^k \cdot p^{(k(k-1)/2)}$. Recalling $p = 1/n^{c/k}$, we get that the probability that G does not have a k-clique is $n^k \cdot n^{-c(k-1)/2} = n^{k - c(k-1)/2}$. Since $c = 2 + \epsilon$, $k - c(k-1)/2 = 1 - \frac{\epsilon}{2}(k-1)$. Hence $n^k \cdot n^{-c(k-1)/2} \leq 2^{-\log n}$ for sufficiently large n and since $k \leq \log n$.

So the probability that either property (1) or (2) does not hold is bounded above by $2^{\log^2 n} \cdot e^{-n^{\delta}} + 2^{-\log^2 n}$ which is below 1 for sufficiently large n. ◀

3.2 Res(s) lower bounds for Bin-Clique$^{\mathsf{n}}_{\mathsf{k}}$

Let $s \geq 1$ be an integer. Call a $\frac{1}{2^{s+1}}$-total assignment to the variables of $\mathsf{Bin\text{-}Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$ an *s-restriction*. A *random s-restriction* for $\mathsf{Bin\text{-}Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$ is an s-restriction obtained by choosing independently in each block i, $\lfloor \frac{1}{2^{s+1}} \log n \rfloor$ variables among $\omega_{i,1}, \ldots, \omega_{i,\log n}$, and setting these uniformly at random to 0 or 1.

Let $s, k \in \mathbb{N}$, $s, k \geq 1$ and let G be graph over nk nodes and k blocks which does not contain a k-clique. Fix $\delta = \frac{1}{24^2}$ and $\mathsf{p}(s) = 2^{(s+1)^2}$ and $\mathsf{d}(s) = (\mathsf{p}(s)s)^s$.

Consider the following property.

▶ **Definition 5** (Property $\mathsf{Clique}(G, s, k)$). *For any $\gamma \geq 2$ and for any γ-restriction ρ, there are no Res(s) refutations of $\mathsf{Bin\text{-}Clique}^{\mathsf{n}}_{\mathsf{k}}(G)\restriction_\rho$ of size less than $n^{\frac{\delta(k-1)}{\mathsf{d}(s)}}$.*

If property $\mathsf{Clique}(G, s, k)$ holds, we immediately have $n^{\Omega(k)}$ size lower bounds for refuting $\mathsf{Bin\text{-}Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$ in $Res(s)$.

▶ **Corollary 6.** *Let s, k be integers, $s \geq 1, k > 1$. Let G be a graph and assume that $\mathsf{Clique}(G, s, k)$ holds. Then there are no Res(s) refutations of $\mathsf{Bin\text{-}Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$ of size smaller that $n^{\delta \frac{k-1}{\mathsf{d}(s)}}$.*

Proof. Choose ρ to be any s-restriction, for $\gamma \geq 1$. The result follows from the previous definition since the shortest refutation of a restricted principle can never be larger than the shortest refutation of the unrestricted principle. ◀

We use the previous corollary to prove lower bounds for $\mathsf{Bin\text{-}Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$ in $\mathsf{Res}(s)$ as long as $s = o(\log\log n)$.

▶ **Theorem 7.** *Let $0 < \epsilon < 1$ be given. Let k be an integer with $k > 1$. Let s be an integer with $1 < s \leq \frac{1}{2}\log\log n$. Then there exists a graph G such that Res(s) refutations of $\mathsf{Bin\text{-}Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$ have size $n^{\Omega(k)}$.*

Proof. Let $1 > \alpha > 0$ and $1 > \beta > 0$ such that $1 - \beta > \alpha(2 + \epsilon)$. By Lemma 4, we can fix $G \sim \mathcal{G}^n_{k,\epsilon}$ such that:

1. all α-transversal sets U are β-extendible;
2. $\mathcal{G}$ does not have a k-clique.

We will prove, by induction on $s = o(\log\log n)$, that property $\mathsf{Clique}(G, s, k)$ does hold. The result then follows by Corollary 6. Lemma 8 is the base case and Lemma 9 the inductive case. ◀

▶ **Lemma 8** (Base Case)**.** $\mathsf{Clique}(G, 1, k)$ *does hold.*

Proof. Fix $\beta = \frac{3}{4}$ and $\alpha = \frac{1}{4(2+\epsilon)} \geq \frac{1}{12}$. Notice that $\mathsf{d}(1) = 16$. Let ρ be a 1-restriction, that is a $\frac{1}{4}$-total assignment. We claim that any Resolution refutation of $\mathsf{Bin\text{-}Clique}^{\mathsf{n}}_{\mathsf{k}}(G)\restriction_\rho$ must have width at least $\frac{k\log n}{24}$. This is a consequence of the extension property which allows Adversary to play against Prover with the following strategy: for each block, while fewer than $\frac{\log n}{2}$ bits are known, Adversary offers Prover a free choice. Once $\frac{\log n}{2}$ bits are set then Adversary chooses an assignment for the remaining bits according to the extension property. Since $\frac{1}{4} + \frac{1}{2} = \frac{3}{4}$, this allows the game to continue until some record has width at least $\frac{\log n}{2} \cdot \frac{k}{12} = \frac{k\log n}{24}$. Size-width tradeoffs for Resolution [8] tells us that minimal size to refute any unsat CNF F is lower bounded by $2^{\frac{(w(\vdash F) - w(F))^2}{16V(F)}}$[1]. In our case $w(F) = 2\log n$ and $V(F) = k\log n$, hence the minimal size required is $\geq 2^{\frac{(\frac{k\log n}{24} - 2\log n)^2}{16k\log n}} = 2^{\frac{\log n(\frac{k}{24}-2)^2}{16k}} = n^{\frac{(\frac{k}{24}-2)^2}{16k}}$. It is not difficult to see that $\frac{(\frac{k}{24}-2)^2}{16k} \geq \frac{(k-1)}{16\cdot 24^2}$. Since $\delta = \frac{1}{24^2}$ and $\mathsf{d}(1) = 16$ the result is proved. ◀

▶ **Lemma 9** (Inductive Case)**.** $\mathsf{Clique}(G, s-1, k)$ *implies* $\mathsf{Clique}(G, s, k)$.

Proof. Recall that we fixed $\mathsf{p}(s) = 2^{(s+1)^2}$ and $\mathsf{d}(s) = (\mathsf{p}(s)s)^s$. Set $L(s) = n^{\frac{\delta(k-1)}{\mathsf{d}(s)}}$ and $\chi(s) = \frac{(s-1)^{s-1}}{s^s 2^{3s^2+s}}$. (Proof of the next claim is postponed after the proof.)

▷ Claim 10. $\ln L(s) = \chi(s)\ln L(s-1)$

We prove the contrapositive of the statement of the Lemma. Assume there is some s-restriction ρ such that there exists a Res(s) refutation π of $\mathsf{Bin\text{-}Clique}^{\mathsf{n}}_{\mathsf{k}}(G)\restriction_\rho$ with size less than $L(s)$. We prove that that there is a $(s-1)$-restriction τ such that there are $\mathsf{Res}(s-1)$ proofs of $\mathsf{Bin\text{-}Clique}^{\mathsf{n}}_{\mathsf{k}}(G)\restriction_\tau$ of size $< L(s-1)$.

Consider the function:

$$f(s,n) = \frac{(1-\chi(s))}{(\ln 2)\,\mathsf{d}(s-1)} - \frac{4}{\delta(k-1)\ln n}.$$

$f(s,n)$ is lower bounded as follows (see the proof after the the proof of this Lemma).

▷ Claim 11. For sufficiently large n and for all $s \geq 2$,

$$f(s,n) > \frac{1}{(\mathsf{p}(s)s)^{s-1}}.$$

[1] According to [25] Th 8.11

Fix the covering number as:

$c = f(s,n)\delta(k-1)\ln n$

Define $r = \frac{c}{s}$ and let us call a *bottleneck* a record R in π whose covering number is $\geq c$. Hence in such a record it is always possible to find r pairwise disjoint s-tuples of literals $T_1 = (\ell_1^1, \ldots, \ell_1^s), \ldots, T_r = (\ell_r^1, \ldots, \ell_r^s)$ such that the $\bigwedge T_i$'s are the terms of the s-DNF forming the record R.

Let σ be a *s-random restriction* on the variables of $\mathsf{Bin\text{-}Clique}_{\mathsf{k}}^{\mathsf{n}}(G)\restriction_\rho$. Let us say that σ *kills a tuple* T if it sets to 0 all literals in T (notice that a record is the negation of a s-DNF) and that T *survives* σ otherwise. And that σ *kills* R if it kills at at least one of the tuples in R. Let Σ_i be the event that T_i survives σ and Σ_R the event that R survives σ. We claim (postponing the proof) that

▷ Claim 12. $\Pr[\Sigma_R] \leq (1 - \frac{1}{\mathsf{p}(s)})^r$.

Consider now the restriction $\tau = \rho\sigma$. This is a $(s-1)$-restriction on the variables of $\mathsf{Bin\text{-}Clique}_{\mathsf{k}}^{\mathsf{n}}(G)$. We argue that in $\pi\restriction_\tau$ there is no bottleneck. Notice that by the union bound the probability that there exists such a record in $\pi\restriction_\tau$, is bounded by

$$\Pr[\exists R \in \pi\restriction_\tau \colon \Sigma_R] \leq |\pi\restriction_\tau|(1 - \frac{1}{\mathsf{p}(s)})^r.$$

We claim that this probability is < 1. Notice that $(1 - \frac{1}{\mathsf{p}(s)})^r \leq e^{-\frac{c}{s\,\mathsf{p}(s)}}$ using the definition of r. So to prove the claim it is sufficient to prove that $|\pi\restriction_\tau| < e^{\frac{c}{\mathsf{p}(s)s}}$ or equivalently that $\ln|\pi\restriction_\tau| < \frac{c}{s\,\mathsf{p}(s)}$. But $\ln|\pi\restriction_\tau| \leq \ln|\pi| = \ln L(s) = \frac{1}{s\,\mathsf{p}(s)}\frac{\delta(k-1)\ln n}{(\mathsf{p}(s)s)^{s-1}}$. Since by Claim 11 $f(s,n) > \frac{1}{(\mathsf{p}(s)s)^{s-1}}$, then $\ln|\pi\restriction_\tau| < \frac{f(s,n)\delta(k-1)\ln n}{s\,\mathsf{p}(s)} = \frac{c}{s\,\mathsf{p}(s)}$, where the last inequality follows by definition of c.

Since in $\pi\restriction_\tau$ there is no bottleneck, by Lemma 1, we can morph $\pi\restriction_\tau$ through the restriction τ to a $\mathsf{Res}(s-1)$ refutation of $\mathsf{Bin\text{-}Clique}_{\mathsf{k}}^{\mathsf{n}}(G)\restriction_\tau$ of size $2^{c+2}\cdot L(s)$. Hence the Lemma is proved arguing that

$$2^{c+2}\cdot L(s) < L(s-1) \tag{4}$$

Since by Claim 10, $\ln L(s) = \chi(s)\ln L(s-1)$, we have the following equivalences:

$$(c+2)\ln 2 + \ln L(s) < \ln L(s-1) \qquad \text{Passing to ln of Eq. 4} \tag{5}$$

$$(c+2)\ln 2 < \ln L(s-1)(1-\chi(s)) \tag{6}$$

$$(f(s,n)\delta(k-1)\ln n + 2)\ln 2 < \frac{\delta(k-1)\ln n}{\mathsf{d}(s-1)}\cdot\frac{(1-\chi(s))}{} \qquad \text{def of } c \text{ and of } L(s-1) \tag{7}$$

$$f(s,n)\delta(k-1)\ln n + 2 < \frac{\delta(k-1)\ln n}{\mathsf{d}(s-1)}\cdot\frac{(1-\chi(s))}{\ln 2} \qquad \text{dividing by } \ln 2 \tag{8}$$

$$f(s,n)\delta(k-1)\ln n < \frac{\delta(k-1)\ln n}{\mathsf{d}(s-1)}\cdot\frac{(1-\chi(s))}{\ln 2} - 2 \qquad \text{subtracting 2} \tag{9}$$

$$f(s,n) < \frac{(1-\chi(s))}{(\ln 2)\,\mathsf{d}(s-1)} - \frac{2}{\delta(k-1)\ln n}. \qquad \text{dividing by } \delta(k-1)\ln n \tag{10}$$

The last line is true since by its definition $f(s,n) = \frac{(1-\chi(s))}{(\ln 2)\,\mathsf{d}(s-1)} - \frac{4}{\delta(k-1)\ln n}$. ◀

Notice that the due to the definition of $L(s)$ the proof can be carried as long as $(s\,\mathsf{p}(s))^s \leq \ln n$ which means $s = o(\log\log n)$.

Proof of Claim 10. Notice that $\mathsf{p}(s-1) = 2^{s^2}$ and that $\mathsf{p}(s) = 2^{s^2}2^{2s+1} = \mathsf{p}(s-1)2^{2s+1}$. Consider the following equalities

$$\ln L(s) = \frac{\delta(k-1)\ln n}{(\mathsf{p}(s)s)^s} \tag{11}$$

$$= \frac{\delta(k-1)\ln n}{(\mathsf{p}(s-1)2^{2s+1})^s s^s} \cdot \frac{(s-1)^{s-1}}{(s-1)^{s-1}} \tag{12}$$

$$= \frac{\delta(k-1)\ln n}{\mathsf{p}(s-1)^{s-1}(s-1)^{s-1}} \cdot \frac{(s-1)^{s-1}}{s^s\,\mathsf{p}(s-1)(2^{2s+1})^s} \tag{13}$$

$$= \frac{\delta(k-1)\ln n}{\mathsf{p}(s-1)^{s-1}(s-1)^{s-1}} \cdot \frac{(s-1)^{s-1}}{s^s\,\mathsf{p}(s-1)2^{2s^2+s}} \tag{14}$$

$$= \frac{\delta(k-1)\ln n}{\mathsf{d}(s-1)} \cdot \frac{(s-1)^{s-1}}{s^s 2^{s^2}2^{2s^2+s}} \tag{15}$$

$$= L(s-1) \cdot \frac{(s-1)^{s-1}}{s^s 2^{3s^2+s}} \tag{16}$$

Notice that $\chi(s) = \frac{(s-1)^{s-1}}{s^s 2^{3s^2+s}}$ so the result follows. ◁

Proof of Claim 11. For $n \to \infty$, $\frac{4}{\delta(k-1)\ln n} \to 0$, so for a sufficiently large n we can ignore the term $\frac{4}{\delta(k-1)\ln n}$. Moreover since $\ln 2 < 1$ we forgot the factor $\frac{1}{\ln 2}$ in $f(s,n)$. We have to show that for all $s \geq 2$

$$\frac{(1-\chi(s))}{(\mathsf{p}(s-1)(s-1))^{s-1}} > \frac{1}{(\mathsf{p}(s)s)^{s-1}}. \tag{17}$$

First we bound the RHS in a convenient form. First since $\frac{1}{s-1} > \frac{1}{s}$ the claim in Eq 17 follows from proving that

$$\frac{(1-\chi(s))}{(\mathsf{p}(s-1)(s-1))^{s-1}} > \frac{1}{(\mathsf{p}(s)(s-1))^{s-1}}. \tag{18}$$

Recall from proof of Claim 10 that $\mathsf{p}(s) = \mathsf{p}(s-1)2^{2s+1}$. Hence we can write the denominator $(\mathsf{p}(s)(s-1))^{s-1}$ of RHS of Eq. 18 as

$$(\mathsf{p}(s)(s-1))^{s-1} = (\mathsf{p}(s-1)(s-1))^{s-1} \cdot (2^{2s+1})^{s-1} \tag{19}$$

$$= (\mathsf{p}(s-1)(s-1))^{s-1} \cdot 2^{2s^2-(s+1)} \tag{20}$$

Hence Eq. 18 follows from proving

$$\frac{(1-\chi(s))}{(\mathsf{p}(s-1)(s-1))^{s-1}} > \frac{1}{(\mathsf{p}(s-1)(s-1))^{s-1} \cdot 2^{2s^2-(s+1)}} \tag{21}$$

Multiplying both sides by $(\mathsf{p}(s-1)(s-1))^{s-1}$ this is equivalent to prove that

$$(1-\chi(s)) > \frac{1}{2^{2s^2-(s+1)}} \tag{22}$$

Which is equivalent to prove that

$$(1-\chi(s)) > \frac{2^{s+1}}{2^{2s^2}} \tag{23}$$

Now we work on a more convenient form of LHS. Recall that

$$\chi(s) = \frac{(s-1)^{s-1}}{s^s 2^{3s^2+s}}$$

so that

$$1 - \chi(s) = \frac{s^s 2^{3s^2+s} - (s-1)^{s-1}}{s^s 2^{3s^2+s}} \tag{24}$$

So Eq 23 can be rewritten as

$$\frac{s^s 2^{3s^2+s} - (s-1)^{s-1}}{s^s 2^{3s^2+s}} > \frac{2^{s+1}}{2^{2s^2}} \tag{25}$$

Multiplying both sides by $s^s 2^{3s^2+s}$ we have the equivalent equation

$$s^s 2^{3s^2+s} - (s-1)^{s-1} > 2^{s+1} s^s 2^{s^2+s} \tag{26}$$

which, dividing both sides by s^s is equivalent to prove

$$2^{3s^2+s} - \frac{(s-1)^{s-1}}{s^s} > 2^{s^2+2s+1} \tag{27}$$

First we claim that $\frac{(s-1)^{s-1}}{s^s} < 1$, which is equivalent tor prove that $(s-1)\ln(s-1) - s\ln(s) < 0$ by passing to logarithms. But $(s-1)\ln(s-1) - s\ln(s) < (s-1)\ln(s-1) - (s-1)\ln(s) < (s-1)\ln(s-1) - (s-1)\ln(s-1) = 0$.

So

$$2^{3s^2+s} - \frac{(s-1)^{s-1}}{s^s} > 2^{s^2+s} - 1$$

and Eq 27 follows from proving that

$$2^{3s^2+s} - 1 \geq 2^{s^2+2s+1} \tag{28}$$

divide both sides by the RHS, which is 2^{s^2+2s+1} so that we want to prove that

$$2^{3s^2+s-(s^2+2s+1)} - \frac{1}{2^{s^2+2s+1}} \geq 1 \tag{29}$$

Again $\frac{1}{2^{s^2+2s+1}} \leq 1$ and $2^{3s^2+s-(2s^2+2s+1)} = 2^{2s^2-(s+1)}$, so that Eq 29 follows from proving that

$$2^{2s^2-(s+1)} - 1 \geq 1 \tag{30}$$

$2^{2s^2-(s+1)}$ is a growing function in s and for $s = 2$ is value is exactly $2^5 = 32 > 2$. Hence it is always true that $2^{s^2-(s+1)} \geq 2$, which proves Eq 29 and hence our Claim. ◁

Proof of Claim 12. Since $T_1, \ldots, T_r$ are tuples in R, then $\Pr[\Sigma_R] \leq \Pr[\Sigma_1 \wedge \ldots \wedge \Sigma_r]$. Moreover $\Pr[\Sigma_1 \wedge \ldots \wedge \Sigma_r] = \prod_{i=1}^{r} \Pr[\Sigma_i | \Sigma_1 \wedge \ldots \wedge \Sigma_{i-1}]$. We will prove that for all $i = 1, \ldots, r$,

$$\Pr[\Sigma_i | \Sigma_1 \wedge \ldots \wedge \Sigma_{i-1}] \leq \Pr[\Sigma_i] \tag{31}$$

Hence the result follows from Lemma 13 which is proving that $\Pr[\Sigma_i] \leq 1 - \frac{1}{\mathsf{p}(s)}$.

By Lemma 14 (i), to prove that Equation 31 holds, we show that $\Pr[\Sigma_i | \neg\Sigma_1 \vee \ldots \vee \neg\Sigma_{i-1}] \geq \Pr[\Sigma_i]$. We claim that for $j \in [r], i \neq j$:

$$\Pr[\Sigma_i | \neg\Sigma_j] \geq \Pr[\Sigma_i] \tag{32}$$

Hence repeated applications of Lemma 14 (ii), prove that $\Pr[\Sigma_i | \neg\Sigma_1 \vee \ldots \vee \neg\Sigma_{i-1}] \geq Pr[\Sigma_i]$.

To prove Equation 32, let $B(T_i)$ be the set of blocks mentioned in T_i. If $B(T_i)$ and $B(T_j)$ are disjoint, then clearly $\Pr[\Sigma_i | \neg\Sigma_j] = \Pr[\Sigma_i]$. When $B(T_i)$ and $B(T_j)$ are not disjoint, we reason as follows: For each $\ell \in B(T_i)$, let T_i^ℓ be the set of variables in T_i mentioning block ℓ. T_i is hence partitioned into $\bigcup_{\ell \in B(T_i)} T_i^\ell$ and hence the event "T_i surviving σ", can be partitioned into the sum of the events that T_i^ℓ survives to σ, for $\ell \in B(T_i)$. Denote by Σ_i^ℓ the event "T_i^ℓ survives σ" and let A=$B(T_i) \cap B(T_j)$ and $B = B(T_i) \setminus (B(T_i) \cap B(T_j))$. The following inequalities holds:

$$\begin{aligned}
\Pr[\Sigma_i | \neg\Sigma_j] &= \Pr[\exists \ell \in B(T_i) : \Sigma_i^\ell | \neg\Sigma_j] && (33)\\
&= \sum_{\ell \in B(T_i)} \Pr[\Sigma_i^\ell | \neg\Sigma_j] && (34)\\
&= \sum_{\ell \in A} \Pr[\Sigma_i^\ell | \neg\Sigma_j] + \sum_{\ell \in B} \Pr[\Sigma_i^\ell | \neg\Sigma_j] && (35)\\
& && (36)
\end{aligned}$$

Since B is disjoint from $B(T_j)$, as for the case above for each $\ell \in B$, $\Pr[\Sigma_i^\ell | \neg\Sigma_j] = \Pr[\Sigma_i^\ell]$. Then:

$$\sum_{\ell \in B} \Pr[\Sigma_i^\ell | \neg\Sigma_j] = \sum_{\ell \in B} \Pr[\Sigma_i^\ell] \tag{37}$$

$$\tag{38}$$

Notice that T_i and T_j are disjoint, hence knowing that some indices in blocks $\ell \in A$ are already chosen to kill T_j, only increase the chances of T_i to survive (since less positions are left in the blocks $\ell \in A$ to potentially kill T_i).

Hence:

$$\sum_{\ell \in A} \Pr[\Sigma_i^\ell | \neg\Sigma_j] \geq \sum_{\ell \in A} \Pr[\Sigma_i^\ell] \tag{39}$$

$$\tag{40}$$

Which proves the claim since:

$$\sum_{\ell \in A} \Pr[\Sigma_i^\ell] + \sum_{\ell \in B} \Pr[\Sigma_i^\ell] = \Pr[\Sigma_i] \tag{41}$$

◁

▶ **Lemma 13.** *Let ρ be a s-random restriction. For all s-tuples S:*

$$\Pr[S \textit{ survives } \rho] \leq 1 - \frac{1}{\mathsf{p}(s)}$$

Proof. Let $T = (\ell_{i_1,j_1}, \ldots, \ell_{i_s,j_s})$ be an s-tuple made of of disjoint literals of $\mathsf{Bin\text{-}Clique}^{\mathsf{n}}_{\mathsf{k}}(G)$. We say that T is *perfect* if all literals are bits of a same block.

Let $\gamma = \frac{1}{2^{s+1}}$. A block with r distinct bits contributes a factor of

$$\frac{\binom{\gamma \log n}{r}}{\binom{\log n}{r}} \cdot \frac{1}{2^r}$$

to the probability that the s-tuple **does not** survive. Expanding the left-hand part of this we obtain

$$\frac{\gamma \log n \cdot \gamma \log n\ -1 \cdots \gamma \log n\ -r+1}{\log n \cdot \log n\ -1 \cdots \log n\ -r+1} = \gamma \frac{\log n}{\log n} \cdot \gamma \frac{\log n\ -\frac{1}{\gamma}}{\log n\ -1} \cdots \gamma \frac{\log n\ -\frac{r}{\gamma}+\frac{1}{\gamma}}{\log n\ -r+1}$$

Next, let us note that

$$1 = \frac{\log n}{\log n} > \frac{\log n\ -\frac{1}{\gamma}}{\log n\ -1} > \cdots > \frac{\log n\ -\frac{r}{\gamma}+\frac{1}{\gamma}}{\log n\ -r+1}$$

The result now follows when we recall that the probability of surviving is maximised when the probability of not surviving is minimised. ◀

▶ **Lemma 14.** *Let A, B, C three events such that $\Pr[A], \Pr[B], \Pr[C] > 0$:*

(i) *If $\Pr[A|\neg B] \geq \Pr[A]$ then $\Pr[A|B] \leq \Pr[A]$;*

(ii) *$\Pr[A|B] \geq \Pr[A]$ and $\Pr[A|C] \geq \Pr[A]$. Then $\Pr[A|B \vee C] \geq \Pr[A]$.*

Proof. For part (i) consider the following equivalences:

$$\begin{array}{lcl}
\Pr[A] & = & \Pr[A|B]\Pr[B] + \Pr[A|\neg B]\Pr[\neg B] \\
\Pr[A] & = & \Pr[A|B]\Pr[B] + \Pr[A|\neg B](1 - \Pr[B]) \\
\Pr[A] & \geq & \Pr[A|B]\Pr[B] + \Pr[A](1 - \Pr[B]) \\
\Pr[A]\Pr[B] & \geq & \Pr[A|B]\Pr[B] \\
\Pr[A] & \geq & \Pr[A|B]
\end{array}$$

For part (ii) consider the following inequalities:

$$\begin{array}{lcl}
\Pr[A|B \vee C] & = & \frac{\Pr[A \wedge (B \vee C)]}{\Pr[B \vee C]} \\
& \geq & \frac{\Pr[A \wedge B]}{\Pr[B \vee C]} + \frac{\Pr[A \wedge C]}{\Pr[B \vee C]} \\
& = & \frac{\Pr[A \wedge B]}{\Pr[B]} \cdot \frac{\Pr[B]}{\Pr[B \vee C]} + \frac{\Pr[A \wedge C]}{\Pr[C]} \cdot \frac{\Pr[C]}{\Pr[B \vee C]} \\
& = & \Pr[A|B] \cdot \frac{\Pr[B]}{\Pr[B \vee C]} + \Pr[A|C] \cdot \frac{\Pr[C]}{\Pr[B \vee C]} \\
& \geq & \Pr[A] \cdot (\frac{\Pr[B] + \Pr[C]}{\Pr[B \vee C]}) \\
& \geq & \Pr[A]
\end{array}$$

◀

4 The weak Pigeonhole Principle

For $n < m$, let $\mathsf{Bin\text{-}PHP}^m_n$ be the binary encoding of the (weak) Pigeonhole Principle. $\mathsf{Bin\text{-}PHP}^m_n$ is a well-known formula and its definition can be found in Section 5. First notice that an analogous of Lemma 2 holds for the Pigeonhole Principle too.

▶ **Lemma 15.** *Suppose there are Resolution refutations of PHP^m_n of size S. Then there are $\mathsf{Res}(\log n)$ refutations of $\mathsf{Bin\text{-}PHP}^m_n$ of size S.*

Let ρ be a partial assignment (a restriction) to the variables of $\mathsf{Bin\text{-}PHP}^m_n$. We call ρ a *t-bit* restriction if ρ assigns t bits of each pigeon $b \in [m]$, i.e. t variables $\omega_{b,i}$ for each pigeon b. Let $v = (i, a)$ be an assignment meaning that pigeon i is assigned to hole a and let $a_1 \dots a_{\log n}$ be the binary representation of a. We say that a restriction ρ is *consistent* with v if for all $j \in [\log n]$, $\sigma(\omega_{i,j})$ is either a_j or not assigned. We denote by $\mathsf{Bin\text{-}PHP}^m_n{\restriction_\rho}$, $\mathsf{Bin\text{-}PHP}^m_n$ restricted by ρ. We will also consider the situation in which an s-bit restriction is applied to some $\mathsf{Bin\text{-}PHP}^m_n{\restriction_\rho}$, creating $\mathsf{Bin\text{-}PHP}^m_n{\restriction_\tau}$, where τ is an $s+t$-bit restriction.

Throughout this section, let $u = u(n,t) := (\log n) - t$. We do not use this shorthand universally, but sometimes where otherwise the notation would look cluttered. We also occasionally write $(\log n) - t$ as $\log n - t$ (note the extra space).

▶ **Lemma 16.** *Let ρ be a t-bit restriction for $\mathsf{Bin\text{-}PHP}^m_n$. Any decision DAG for $\mathsf{Bin\text{-}PHP}^m_n{\restriction_\rho}$ must contain a record which mentions $\frac{n}{2^t}$ pigeons.*

Proof. Let Adversary play in the following fashion. While some pigeon is not mentioned at all, let him give Prover a free choice to answer any one of its bits as true or false. Once a pigeon is mentioned once, then let Adversary choose a hole for that pigeon by choosing some assignment for the remaining unset bits (we will later need to prove this is always possible). Whenever another bit of an already mentioned pigeon is queried, then Adversary will answer consistently with the hole he has chosen for it. Only once all of a pigeon's bits are forgotten (not including those set by ρ), will Adversary forget the hole he assigned it.

It remains to argue that Adversary must force Prover to produce a record of width $\geq \frac{n}{2^{t+1}}$ and for this it suffices to argue that Adversary can remain consistent with $\mathsf{Bin\text{-}PHP}^m_n{\restriction_\rho}$ up until the point that such a record exists. For that it is enough to show that there is always a hole available for a pigeon for which Adversary gave its only currently questioned bit as a free choice (but for which ρ has already assigned some bits).

The current record is assumed to have fewer than $\frac{n}{2^t}$ literals and therefore must mention fewer than $\frac{n}{2^t}$ pigeons, each of which Adversary already assigned a hole. Each hitherto unmentioned pigeon that has just been given a free choice has $\log n - t$ bits which corresponds to $\frac{n}{2^t}$ holes. Since we have assigned fewer than $\frac{n}{2^t}$ pigeons to holes, one of these must be available, and the result follows. ◀

Let $\xi(s)$ satsify $\xi(1) = 1$ and $\xi(s) = \xi(s-1) + 1 + s$. Note that $\xi(s) = \Theta(s^2)$.

▶ **Definition 17** (Property $\mathsf{PHP}(s,t)$)**.** *Let $s, t \geq 1$. For any t-bit restriction ρ to $\mathsf{Bin\text{-}PHP}^m_n$, there are no $\mathsf{Res}(s)$ refutations of $\mathsf{Bin\text{-}PHP}^m_n{\restriction_\rho}$ of size smaller than $e^{\frac{n}{4^{\xi(s)+1}s!2^t u^{\xi(s)}}}$.*

▶ **Theorem 18.** *Let ρ be a t-bit restriction for $\mathsf{Bin\text{-}PHP}^m_n$. Any decision DAG for $\mathsf{Bin\text{-}PHP}^m_n{\restriction_\rho}$ is of size $2^{\Omega(\frac{n}{\log n})}$ (indeed, asymptotically of size $\geq e^{\frac{n}{2^{t+2}u}}$).*

Proof. Call a *bottleneck* a record in the decision DAG that mentions $\frac{n}{2^{t+1}}$ pigeons. Now consider a random restriction that picks for each pigeon one bit uniformly at random and sets this to 0 or 1 with equal probability. The probability that a bottleneck survives (is not falsified by) the random restriction is no more than

$$\left(\frac{u-1}{u} + \frac{1}{2u}\right)^{\frac{n}{2^{t+1}}} = \left(1 - \frac{1}{2u}\right)^{u \cdot \frac{n}{2^{t+1}u}} \leq \frac{1}{e^{\frac{n}{2^{t+2}u}}},$$

since $e^{-x} = \lim_{m\to\infty}(1 - x/m)^m$ and indeed $e^{-x} \geq (1 - x/m)^m$ when $x, m \geq 1$.

Now suppose for contradiction that we have fewer than $e^{\frac{n}{2^{t+2}u}}$ bottlenecks in a decision DAG for $\mathsf{Bin\text{-}PHP}^m_n{\restriction_\rho}$. By the union bound there is a random restriction that kills all bottlenecks and this leaves a decision DAG for some $\mathsf{Bin\text{-}PHP}^m_n{\restriction_\sigma}$, where σ is a $(t+1)$-bit restriction for $\mathsf{Bin\text{-}PHP}^m_n$. However, we know from Lemma 16 that such a refutation must involve a record mentioning $\frac{n}{2^{t+1}}$ pigeons. This is now the desired contradiction. ◀

Note that the previous theorem could have been proved, like Lemma 8, by the size-width trade-off. However, the method of random restrictions used here could not be easily applied there, due to the randomness of G.

▶ **Corollary 19.** *Property* $\mathsf{PHP}(1,t)$ *holds, for each* $t < \log n$.

Note that, $\mathsf{PHP}(1,t)$ yields only trivial bounds as t approaches $\log n$.

Let $(\ell_{i_1,j_1},\ldots,\ell_{i_s,j_s})$ be an s-tuple made of disjoint literals of $\mathsf{Bin\text{-}PHP}^m_n \restriction_\rho$. We say that a tuple is *perfect* if all literals come from the same pigeon.

▶ **Lemma 20.** *Let* s *be an integer,* $s \geq 1$ *and* $s+t < \log n$. *Let* σ *be a random* s*-bit restriction over* $\mathsf{Bin\text{-}PHP}^m_n \restriction_\rho$ *where* ρ *is itself some* t*-bit restriction over* $\mathsf{Bin\text{-}PHP}^m_n$. *Let* T *be a perfect* s*-tuple of* $\mathsf{Bin\text{-}PHP}^m_n \restriction_\rho$. *Then for all* s*-tuples* S*:*

$$\Pr[T \textit{ survives } \sigma] \geq \Pr[S \textit{ survives } \sigma].$$

and so $\Pr[S \textit{ survives } \sigma] \leq 1 - \frac{1}{u^s}$.

Proof. A pigeon with r distinct bits contributes to not surviving a factor of

$$\frac{s}{\log n - t} \cdot \frac{s-1}{\log n - t - 1} \cdots \frac{s-r+1}{\log n - t - r + 1} \cdot \frac{1}{2^r}.$$

Noting that

$$\frac{s}{\log n - t} \cdot \frac{s-1}{\log n - t - 1} \cdots \frac{1}{\log n - t - s + 1} \cdot \frac{1}{2^r} > \frac{1}{u^s}$$

the result now follows when we recall that the probability of surviving is maximised when the probability of not surviving is minimised. ◀

▶ **Theorem 21.** *Let* $s > 1$ *and* $s + t < \log n$. *Then,* $\mathsf{PHP}(s-1, s+t)$ *implies* $\mathsf{PHP}(s,t)$.

Proof. We proceed by contraposition. Assume there is some t-bit restriction ρ so that there exists a $\mathsf{Res}(s)$ refutation π of $\mathsf{Bin\text{-}PHP}^m_n \restriction_\rho$ with size less than $e^{\frac{n}{4^{\xi(s)+1} \cdot s! 2^t u^{\xi(s)}}}$.

Call a *bottleneck* a record that has covering number $\geq \frac{n}{4^{\xi(s)} \cdot (s-1)! 2^t u^{\xi(s-1)}}$. In such a record, by dividing by s and u, it is always possible to find $r := \frac{n}{4^{\xi(s)} s! 2^t u^{\xi(s-1)+1}}$ s-tuples of literals $(\ell^1_1,\ldots,\ell^s_1),\ldots,(\ell^1_r,\ldots,\ell^s_r)$ so that each s-tuple is a clause in the record and no pigeon appearing in the ith s-tuple also appears in the jth s-tuple (when $i \neq j$). This important independence condition plays a key role. Now consider a random restriction that, for each pigeon, picks uniformly at random s bit positions and sets these to 0 or 1 with equal probability. The probability that the ith of the r s-tuples survives the restriction is maximised when each variable among the s describes a different pigeon (by Lemma 20) and is therefore bound above by

$$\left(1 - \frac{1}{u^s}\right)$$

whereupon

$$\left(1 - \frac{1}{u^s}\right)^{\frac{n}{4^{\xi(s)} s! 2^t u^{\xi(s-1)+1}}} = \left(1 - \frac{1}{u^s}\right)^{\frac{nu^s}{4^{\xi(s)} s! 2^t u^{(\xi(s-1)+1+s)}}}$$

which is $\leq 1/e^{\frac{n}{4^{\xi(s)+1}s!\cdot 2^t u^{\xi(s)}}}$. Supposing therefore that there are fewer than $e^{\frac{n}{4^{\xi(s)+1}s!\cdot 2^t u^{\xi(s)}}}$ bottlenecks, one can deduce a random restriction that kills all bottlenecks. What remains after doing this is a $\mathsf{Res}(s)$ refutation of some $\mathsf{Bin\text{-}PHP}^m_n\restriction_\sigma$, where σ is a $s+t$-bit restriction, which moreover has covering number $< \frac{n}{4^{\xi(s)}\cdot(s-1)!2^t u^{\xi(s-1)}}$. But if the remaining $\mathsf{Res}(s)$ refutation is of size $< e^{\frac{n}{4^{\xi(s)+1}s!\cdot 2^t u^{\xi(s)}}}$ then, from Lemma 1, it would give a $\mathsf{Res}(s-1)$ refutation of size

$$< e^{\frac{n}{4^{\xi(s)}\cdot(s-1)!2^t u^{\xi(s-1)}}} \cdot e^{\frac{n}{4^{\xi(s)+1}s!\cdot 2^t u^{\xi(s)}}} = e^{\frac{n}{4^{\xi(s)}\cdot(s-1)!2^t u^{\xi(s-1)}}(1+\frac{1}{4su^{s+1}})}$$

$$< e^{\frac{2n}{4^{\xi(s)}\cdot(s-1)!2^t u^{\xi(s-1)}}} < e^{\frac{n}{4^{\xi(s)}\cdot(s-1)!2^{t-1} u^{\xi(s-1)}}} < e^{\frac{n}{4^{\xi(s)-s}\cdot(s-1)!2^{s+t} u^{\xi(s-1)}}},$$

since $4^s > 2^{s+1}$, which equals $e^{\frac{n}{4^{\xi(s-1)+1}\cdot(s-1)!2^{s+t} u^{\xi(s-1)}}}$ in contradiction to the inductive hypothesis. ◀

▶ **Theorem 22.** *Fix $\lambda, \mu > 0$. Any refutation of $\mathsf{Bin\text{-}PHP}^m_n$ in $\mathsf{Res}(\sqrt{2}\log^{\frac{1}{2+\lambda}} n)$ is of size $2^{\Omega(n^{1-\mu})}$.*

Proof. First, let us claim that $\mathsf{PHP}(\sqrt{2}\log^{\frac{1}{2+\lambda}} n, 0)$ holds (and this would hold also at $\lambda = 0$). Applying Theorem 21 gives ℓ such that $\frac{\ell(\ell+1)}{2} < \log n$. Noting $\frac{\ell^2}{2} < \frac{\ell(\ell+1)}{2}$, the claim follows.

Now let us look at the bound we obtain by plugging in to $e^{\frac{n}{4^{\xi(s)+1}\cdot s!2^t u^{\xi(s)}}}$ at $s = \sqrt{2}\log^{\frac{1}{2+\lambda}} n$ and $t = 0$. We recall $\xi(s) = \Theta(s^2)$. It follows, since $\lambda > 0$, that each of $4^{\xi(s)+1}$, $s!$ and $\log^{\xi(s)} n$ is $o(n^\mu)$. The result follows. ◀

4.1 The treelike case

Concerning the Pigeonhole Principle, we can prove that the relationship between PHP and Bin-PHP is different for treelike Resolution from general Resolution. In particular, for very weak Pigeonhole Principles, we know the binary encoding is harder to refute in general Resolution; whereas for treelike Resolution it is the unary encoding which is the harder.

▶ **Theorem 23.** *The treelike Resolution complexity of $\mathsf{Bin\text{-}PHP}^m_n$ is $2^{\Theta(n)}$.*

Proof. For the lower bound, one can follow the proof of Lemma 16 with $t = 0$ and finds n free choices on each branch of the tree. Following the method of Riis [34], we uncover a subtree of the decision tree of size 2^n.

For an upper bound of 2^{2n} we pursue the following strategy. First we choose some $n+1$ pigeons to question. We then question all of them on their first bit and separate these into two sets T_1 and F_1 according to whether this was answered true or false. If n is a power of 2, choose the larger of these two sets (if they are the same size then choose either). If n is not a power of two, the matter is mildly complicated, and one must look at how many holes are available with the first bit set to 1, say h^1_1; versus 0, say h^0_1. At least one of $|T_1| > h^1_1$ or $|F_1| > h^0_1$ must hold and one can choose between T_1 and F_1 correspondingly. Now question the second bit, producing two sets T_2 and F_2, and iterate this argument. We will reach a contradiction in $\log n$ iteration since we always choose a set of maximal size. The depth of our tree is bound above by $n + \frac{n}{2} + \frac{n}{4} + \cdots < 2n$ and the result follows. ◀

5 Contrasting unary and binary encodings

To work with a more general theory in which to contrast the complexity of refuting the binary and unary versions of combinatorial principles, following Riis [34] we consider principles which are expressible as first order formulae with no finite model in Π_2-form, i.e. as $\forall\vec{x}\exists\vec{w}\varphi(\vec{x},\vec{w})$ where $\varphi(\vec{x},\vec{y})$ is a formula built on a family of relations $\vec{R}$. For example the *Ordering*

Principle, which states that a finite partial order has a maximal element is one of such principle. Its negation can be expressed in Π_2-form as:

$$\forall x, y, z \exists w \; \neg R(x,x) \wedge (R(x,y) \wedge R(y,z) \rightarrow R(x,z)) \wedge R(x,w).$$

This can be translated into a unsatisfiable CNF $\mathsf{OP_n}$ using a *unary encoding* of the witness, as shown below. In Definition 25 we explain how to generate a binary encoding $\mathsf{Bin\text{-}C_n}$ from any combinatorial principle $\mathsf{C_n}$ expressible as a first order formulae in Π_2-form with no finite models and whose unary encoding we denote by $\mathsf{Un\text{-}C_n}$. For example $\mathsf{Bin\text{-}OP_n}$ would be the conjunction of the clauses below.

$\mathsf{OP_n}$: *Unary encoding*

$$\begin{array}{ll} \overline{v}_{x,x} & x \in [n] \\ \overline{v}_{x,y} \vee \overline{v}_{y,z} \vee v_{x,z} & x,y,z \in [n] \\ \bigvee_{i \in [n]} v_{x,i} & x \in [n] \end{array}$$

$\mathsf{Bin\text{-}OP_n}$: *Binary encoding*

$$\begin{array}{ll} \overline{\nu}_{x,x} & x \in [n] \\ \overline{\nu}_{x,y} \vee \overline{\nu}_{y,z} \vee \nu_{x,z} & x,y,z \in [n] \\ \bigvee_{i \in [\log n]} \omega_{x,i}^{1-a_i} \vee \nu_{x,a} & x,a \in [n] \\ a_1 \ldots a_{\log n} & \text{binary representation of } a \end{array}$$

$$\omega_{x,j}^{a_j} = \begin{cases} \omega_{x,j} & a_j = 1 \\ \overline{\omega}_{x,j} & a_j = 0 \end{cases}$$

As a second example we consider the Pigeonhole Principle which states that a total mapping from $[m]$ to $[n]$ has necessarily a collision when m and n are integers with $m > n$. Following Riis [34], for $m = n+1$, the negation of its relational form can be expressed as a Π_2-formula as

$$\forall x, y, z \exists w \; \neg R(x,0) \wedge (R(x,z) \wedge R(y,z) \rightarrow x = y) \wedge R(x,w)$$

and its usual unary and binary propositional encoding are:

PHP : *Unary encoding*

$$\begin{array}{ll} \bigvee_{j=1}^{n} v_{i,j} & i \in [m] \\ \overline{v}_{i,j} \vee \overline{v}_{i',j} & i, \neq i' \in [m], j \in [n] \end{array}$$

$\mathsf{Bin\text{-}PHP}$: *Binary encoding*

$$\bigvee_{j=1}^{\log n} \overline{\omega}_{i,j} \vee \bigvee_{j=1}^{\log n} \overline{\omega}_{i',j} \quad i \neq i' \in [m]$$

Notice that in the case of Pigeonhole Principle, the existential witness w to the type *pigeon* is of the distinct type *hole*. Furthermore, pigeons only appear on the left-hand side of atoms $R(x,z)$ and holes only appear on the right-hand side. For the Ordering Principle instead, the transitivity axioms effectively enforce the type of y appears on both the left- and right-hand side of atoms $R(x,z)$. This account for why, in the case of the Pigeonhole Principle, we did not need to introduce any new variables to give the binary encoding, yet for the Ordering Principle a new variable w appears.

5.1 Binary encodings of principles involving total comparison

We will now argue that the proof complexity in Resolution of principles involving total comparison will not increase significantly (by more than a polynomial factor) when shifting from the unary encoding to the binary encoding. *Total comparison* is here indicated by the axioms $v_{i,j} \oplus v_{j,i}$, where $\oplus$ indicates XOR, for each $i \neq j$. It follows that it does not make sense to consider the binary encoding of such principles in the search for strong lower bounds. Examples of natural principles involving total comparison include the totally ordered variant of the Ordering Principle (known to be polynomially refutable in Resolution [14]) as well as all of its unary relativisations (which can be exponentially hard for any $\mathrm{Res}(s)$ [17]).

Let $\mathsf{TC\text{-}Prin}$ be some Π_2 first-order principle involving relations of arity no more than 2. Let $n \in \mathbb{N}$ and discover $\mathsf{TC\text{-}Prin}(n)$ with variables $v_{i,j}$, for $i,j \in [n]$, of arity 2, including

axioms of total comparison: $v_{i,j} \oplus v_{j,i}$, for each $i \neq j$. There may additionally be unary variables, of the form u_i, for $i \in [n]$, but no further variables of other arity. Let Un-TC-Prin(n) have axioms $v_{i,1} \vee \ldots \vee v_{i,n}$, for each $i \in [n]$ (for the Ordering Principle this would most naturally correspond to the variant stating a finite total order has a maximal element). To make our translation to the binary encoding, we tacitly assume n is a power or 2. When this is not the case, we need clauses forbidding certain evaluations, and we defer this treatment to Section 5.2. Let Bin-TC-Prin(n) have corresponding variables $\omega_{i,\ell}$ for $i \in [n], \ell \in [\log n]$, where $v_{i,j}$ from the unary encoding semantically corresponds to the conjunction $(\omega_{i,1}^{a_1} \wedge \ldots \wedge \omega_{i,\log n}^{a_{\log n}})$, where

$$\omega_{i,p}^{a_p} = \begin{cases} \omega_{i,p} & \text{if } a_p = 1 \\ \overline{\omega}_{i,p} & \text{if } a_p = 0 \end{cases}$$

with $a_1 \cdots a_{\log n}$ being the binary representation of j. The unary variables stay as they are. From this, the axioms of Bin-TC-Prin(n), including total comparison, can be canonically calculated from the corresponding axioms of Un-TC-Prin(n) as explained in Section 5.2 in Defintion 25. Note that the large disjunctive clauses of Un-TC-Prin(n), that encode the existence of the witness, disappear completely in Bin-TC-Prin(n).

▶ **Lemma 24.** *Suppose there is a Resolution refutation of* Un-TC-Prin(n) *of size* $S(n)$*. Then there is a Resolution refutation of* Bin-TC-Prin(n) *of size at most* $n^2 \cdot S(n)$*.*

Proof. Take a decision DAG π for Un-TC-Prin(n) and consider the point at which some variable $v_{i,j}$ is questioned. Each node in π will be expanded to a small tree in π', which will be a decision DAG for Bin-TC-Prin(n). The question "$v_{i,j}$?" in π will become a sequence of $2 \log n$ questions on variables $\omega_{i,1}, \ldots, \omega_{i,\log n}, \omega_{j,1}, \ldots, \omega_{j,\log n}$, giving rise to a small tree of size $2^{2 \log n} = n^2$ questions in π'. Owing to total comparison, many of the branches of this mini-tree must end in contradiction. Indeed, many of their leaves would imply the impossible $\neg v_{i,j} \wedge \neg v_{j,i}$, while precisely one would imply the impossible $v_{i,j} \wedge v_{j,i}$ (see Figure 1 for an example). Those that don't will always have a sub-branch labelled by $(\omega_{i,1}^{a_1} \wedge \ldots \wedge \omega_{i,\log n}^{a_{\log n}})$, where

$$\omega_{i,p}^{a_p} = \begin{cases} \omega_{i,p} & \text{if } a_p = 1 \\ \overline{\omega}_{i,p} & \text{if } a_p = 0 \end{cases}$$

with $a_1 \cdots a_{\log n}$ being the binary representation of j; **or** $(\omega_{j,1}^{b_1} \wedge \ldots \wedge \omega_{j,\log n}^{b_{\log n}})$, where

$$\omega_{j,p}^{b_p} = \begin{cases} \omega_{j,p} & \text{if } b_p = 1 \\ \overline{\omega}_{j,p} & \text{if } b_p = 0 \end{cases}$$

with $b_1 \cdots b_{\log n}$ being the binary representation of i. By forgetting information along these branches and unifying branches with the same labels of their sub-branches, we are left with precisely these two outcomes, corresponding to "$v_{i,j}$" and "$\neg v_{i,j}$", which is "$v_{j,i}$". Indeed, this is the crux, $\neg v_{i,j}$ being equivalent to $v_{j,i}$, and thus being expressible as some conjunction of variables $\omega_{j,p}^{b_p}$. Thus, π gives rise to π' of size $n^2 \cdot S(n)$ and the result follows. ◀

5.2 Binary versus unary encodings in general

Let $\mathsf{C_n}$ be some combinatorial principle expressible as a first-order Π_2-formula F of the form $\forall \vec{x} \exists \vec{w} \varphi(\vec{x}, \vec{w})$ where $\varphi(\vec{x}, \vec{w})$ is a quantifier-free formula built on a family of relations $\vec{R}$. Following Riis [34] we restrict to the class of such formulae having no finite model.

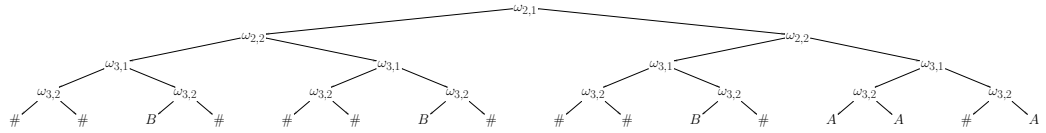


Figure 1 Example converting the question $v_{2,3}$? from a Resolution refutation of Un-TC-Prin(n) to a small tree in a refutation of Bin-TC-Prin(n). The variables $\omega_{2,1}, \omega_{2,2}, \omega_{3,1}, \omega_{3,2}$ are questioned in order. The left-hand and right-hand branches correspond to false and true, respectively. Note that 2 and 3 are 10 and 11 in binary, respectively. Thus, $v_{2,3}$ is equivalent to $\omega_{2,1} \wedge \omega_{2,2}$ (labelled A at the leaves) and $v_{3,2}$ is equivalent to $\omega_{3,1} \wedge \overline{\omega}_{3,2}$ (labelled B at the leaves). The remaining leaves contradict the total comparison clauses (including one that would be labelled both A and B).

Let Un-$\mathsf{C_n}$ be the standard unary (see Riis in [34]) CNF propositional encoding of F. For each set of first-order variables $\vec{a} := \{x_1, \ldots, x_k\}$ of (first order) variables, we consider the propositional variables $v_{x_{i_1}, x_{i_2}, \ldots, x_{i_k}}$ (which we abbreviate as $v_{\vec{a}}$) whose semantics are to capture at once the value of variables in $\vec{a}$ if they appear in some relation in φ. For easiness of description we restrict to the case where F is of the form $\forall \vec{x} \exists w \varphi(\vec{x}, w)$, i.e. $\vec{w}$ is a single variable w. Hence the propositional variables of Un-$\mathsf{C_n}$ are of the type $v_{\vec{a}}$ for $\vec{a} \subseteq \vec{x}$ (type 1 variables) and/or of the type $v_{\vec{x}w}$ for $w \in \vec{w}$ (type 2 variables) and which we denote by simply v_w, since each existential variable in F depends always on all universal variables. Notice that we consider the case of $F = \forall \vec{x} \exists w \varphi(\vec{x}, w)$, since the generalisation to higher arity is clear as each witness $w \in \vec{w}$ may be treated individually.

▶ **Definition 25** (Canonical form of Bin-$\mathsf{C_n}$). *Let* $\mathsf{C_n}$ *be a combinatorial principle expressible as a first-order formula* $\forall \vec{x} \exists w \varphi(\vec{x}, w)$ *with no finite models. Let* Un-$\mathsf{C_n}$ *be its unary propositional encoding. Let* $2^{r-1} < n \leq 2^r \in \mathbb{N}$ $(r = \lceil \log n \rceil)$. *The binary encoding* Bin-$\mathsf{C_n}$ *of* C *is defined as follows:*

The ***variables*** *of* Bin-$\mathsf{C_n}$ *are defined from variables of* Un-$\mathsf{C_n}$ *as follows:*

1. *For each variable of type 1* $v_{\vec{a}}$, *for* $\vec{a} \subseteq \vec{x}$, *we use a variable* $\nu_{\vec{x}}$, *for* $\vec{a} \subseteq \vec{x}$, *and*
2. *For each variable of type 2* v_w, *we have* r *variables* $\omega_1, \ldots \omega_r$, *where we use the convention that if* $z_1 \ldots z_r$ *is the binary representation of* w, *then*

$$\omega_j^{z_j} = \begin{cases} \omega_j & z_j = 1 \\ \overline{\omega}_j & z_j = 0 \end{cases}$$

so that v_w *can be represented using binary variables by the clause* $(\omega_1^{1-z_1} \vee \ldots \vee \omega_r^{1-z_r})$

The ***clauses*** *of* Bin-$\mathsf{C_n}$ *are defined form the clauses of* Un-$\mathsf{C_n}$ *as follows:*

1. *If* $C \in$ Un-$\mathsf{C_n}$ *contains only variables of type 1,* $v_{\vec{b}_1}, \ldots, v_{\vec{b}_k}$, *hence* C *is mapped as follows*

$$C := \bigvee_{j=1}^{k_1} v_{\vec{b}_j} \vee \bigvee_{j=1}^{k_2} \overline{v}_{\vec{c}_j} \quad \mapsto \quad \bigvee_{j=1}^{k_1} \nu_{\vec{b}_j} \vee \bigvee_{j=1}^{k_2} \overline{\nu}_{\vec{c}_j}$$

2. *If* $C \in$ Un-$\mathsf{C_n}$ *contains type 1 and type 2 variables, it is mapped as follows:*

$$C := v_w \vee \bigvee_{j=1}^{k_1} v_{\vec{c}_j} \vee \bigvee_{l=1}^{k_2} \overline{v}_{\vec{d}_j} \quad \mapsto \quad \left(\bigvee_{i \in [r]} \omega_i^{1-z_i}\right) \vee \bigvee_{j=1}^{k_1} \nu_{\vec{c}_j} \vee \bigvee_{l=1}^{k_2} \overline{\nu}_{\vec{d}_j}$$
$$C := \overline{v}_w \vee \bigvee_{j=1}^{k_1} v_{\vec{c}_j} \vee \bigvee_{l=1}^{k_2} \overline{v}_{\vec{d}_j} \quad \mapsto \quad \left(\bigvee_{i \in [r]} \omega_i^{z_i}\right) \vee \bigvee_{j=1}^{k_1} \nu_{\vec{c}_j} \vee \bigvee_{l=1}^{k_2} \overline{\nu}_{\vec{d}_j}$$

where $\vec{c}_j, \vec{d}_l \subseteq \vec{x}$ *and where* $z_1, \ldots, z_r$ *is the binary representation of* w.

3. *If* $n \neq 2^r$, *then, for each* $n < a \leq 2^r$ *we need clauses*

$$\omega_1^{1-a_1} \vee \ldots \vee \omega_r^{1-a_r}$$

where $a_1, \ldots, a_r$ *is the binary representation of* a.

Getting short proofs for the binary version Bin-$\mathsf{C_n}$ in Res($\log n$) form short Res(1) proofs of the unary version Un-$\mathsf{C_n}$ is possible also in the general case.

▶ **Lemma 26.** *Let* $\mathsf{C_n}$ *be a combinatorial principle expressible as a first-order formula* $\forall\vec{x}\exists\vec{w}\varphi(\vec{x},\vec{w})$ *with no finite models. Let* Un-$\mathsf{C_n}$ *and* Bin-$\mathsf{C_n}$ *be respectively the unary and binary propositional encoding. Let* $n \in \mathbb{N}$*. If there is a size* S *refutation for* Un-$\mathsf{C_n}$ *in* Res(1)*, then there is a size* S *refutation for* Bin-$\mathsf{C_n}$ *in* Res($\log n$)

Proof Sketch. Where the decision DAG for Un-$\mathsf{C_n}$ questions some variable $v_{\vec{a},b}$, the decision branching $\log n$-program questions instead $(\omega_{\vec{a},1}^{1-z_1} \vee \ldots \vee \omega_{\vec{a},\log n}^{1-z_{\log n}})$ where the out-edge marked true in the former becomes false in the latter, and vice versa. What results is indeed a decision branching $\log n$-program for Bin-$\mathsf{C_n}$, and the result follows. ◀

As one can easily notice reading Subsection 1.3, the binary version Bin-PHP of the Pigeonhole Principle we displayed there, is different from the one we would get applying the canonical transformation of Definition 5.2. However, we can easily and efficiently move between these versions in Resolution (we leave the proof to the reader below), and the version we have chosen is easier to handle, having fewer variables.

▶ **Lemma 27.** *The two versions of the binary Pigeonhole Principle (*Bin-PHP *and the one arising from Definition 5.2 to* PHP*) are linearly equivalent in Resolution.*

5.3 Binary encodings of principles versus their Unary functional encodings

Recall the unary functional encoding of a combinatorial principle C, denoted Un-Fun-C(n), replaces the big clauses from Un-C(n), of the form $v_{i,1} \vee \ldots \vee v_{i,n}$, with $v_{i,1} + \ldots + v_{i,n} = 1$, where addition is made on the natural numbers. This is equivalent to augmenting the axioms $\neg v_{i,j} \vee \neg v_{i,k}$, for $j \neq k \in [n]$.

▶ **Lemma 28.** *Suppose there is a Resolution refutation of* Bin-C(n) *of size* $S(n)$*. Then there is a Resolution refutation of* Un-Fun-C(n) *of size at most* $n^2 \cdot S(n)$*.*

Proof. Take a decision DAG π' for Bin-C(n), where w.l.o.g. n is even, and consider the point at which some variable $\nu'_{i,j}$ is questioned. Each node in π' will be expanded to a small tree in π, which will be a decision DAG for Un-Fun-C(n). The question "$\nu'_{i,j}$?" in π will become a sequence of questions $v_{i,1}, \ldots, v_{i,n}$ where we stop the small tree when one of these is answered true, which must eventually happen. Suppose $v_{i,k}$ is true. If the jth bit of k is 1 we ask now all $v_{i,b_1}, \ldots, v_{i,b_{\frac{n}{2}}}$, where $b_1, \ldots, b_{\frac{n}{2}}$ are precisely the numbers in $[n]$ whose jth bit is 0. All of these must be false. Likewise, if the jth bit of k is 0 we ask all $v_{i,b_1}, \ldots, v_{i,b_{\frac{n}{2}}}$, where $b_1, \ldots, b_{\frac{n}{2}}$ are precisely the numbers whose jth bit is 1. All of these must be false. We now unify the branches on these two possibilities, forgetting any intermediate information. (To give an example, suppose $j = 2$. Then the two outcomes are $\neg v_{i,1} \wedge \neg v_{i,3} \wedge \ldots \wedge \neg v_{i,n-1}$ and $\neg v_{i,2} \wedge \neg v_{i,4} \wedge \ldots \wedge \neg v_{i,n}$.) Thus, π' gives rise to π of size $n^2 \cdot S(n)$ and the result follows. ◀

5.4 The Ordering Principle in binary

Recall the Ordering Principle specified in Π_2 first-order logic

$$\forall x, y, z \exists w \ \neg R(x,x) \wedge (R(x,y) \wedge R(y,z) \rightarrow R(x,z)) \wedge R(x,w)$$

with propositional translation to the binary encoding of witnesses, Bin-OP$_n$, as follows.

$$\begin{array}{ll} \overline{\nu}_{x,x} & x \in [n] \\ \overline{\nu}_{x,y} \vee \overline{\nu}_{y,z} \vee \nu_{x,z} & x,y,z \in [n] \\ \bigvee_{i \in [\log n]} \omega_{x,i}^{1-a_i} \vee \nu_{x,a} & x,a \in [n] \end{array}$$

where

$$\omega_{i,j}^{a_j} = \begin{cases} \omega_{i,j} & \text{if } a_j = 1 \\ \overline{\omega}_{i,j} & \text{if } a_j = 0 \end{cases}$$

and $a_1 \dots a_{\log n}$ is the binary representation of a.

▶ **Lemma 29.** Bin-OP$_n$ *has refutations in Resolution of polynomial size.*

Proof. We follow the well-known proof for the unary version of the Ordering Principle, from [36]. Consider the domain to be $[n] = \{1, \dots, n\}$. At the ith stage of the decision DAG we will find a maximal element, ordered by R, among $[i] = \{1, \dots, i\}$. That is, we will have a record of the *special* form

$$\overline{\nu}_{j,1} \wedge \dots \wedge \overline{\nu}_{j,j-1} \wedge \overline{\nu}_{j,j+1} \wedge \dots \wedge \overline{\nu}_{j,i}$$

for some $j \in [i]$. The base case $i = 1$ is trivial. Let us explain the inductive step. From the displayed record above we ask the question $\nu_{j,i+1}$? If $\nu_{j,i+1}$ is true, then ask the sequence of questions $\nu_{i+1,1}, \dots, \nu_{i+1,i}$, all of which must be false by transitivity. Now, by forgetting information, we uncover a new record of the special form. Suppose now $\nu_{j,i+1}$ is false. Then we equally have a new record again in the special form. Let us consider the size of our decision tree so far. There are n^2 nodes corresponding to special records and navigating between special records involves a path of length n, so we have a DAG of size n^3. Finally, at $i = n$, we have a record of the form

$$\overline{\nu}_{j,1} \wedge \dots \wedge \overline{\nu}_{j,j-1} \wedge \overline{\nu}_{j,j+1} \wedge \dots \wedge \overline{\nu}_{j,n}.$$

Now we expand a tree questioning the sequence $w_{j,1}, \dots, w_{j,\log n}$, and discover each leaf labels a contradiction of the clauses of the final type. We have now added $n \cdot 2^{\log n}$ nodes, so our final DAG is of size at most $n^3 + n^2$. ◀

▶ **Theorem 30.** Bin-OP$_n$ *has poly size resolution refutations in* Res(1).

6 Final remarks

Various questions are left unanswered in our exposition. Primarily, there is the question as to the optimality of our lower bounds for the binary encodings of k-Clique and the (weak) Pigeonhole Principle. In terms of the strongest refutation system Res(s) (largest s) for which we can prove superpolynomial bounds, then it is not hard to see that our method can go no further than $s = \Theta(\log \log n)$ for the former, and $s = o(\log^{1/2} n)$ for the latter. This is because we run out of space with the random restrictions as they become nested in the induction. We have no reason, however to think that our results are truly optimal, only that another method is needed to improve them.

Similarly, one might ask whether converses to our lemmas might hold. For example, to Lemmas 24 and 26. In these cases, we do not know about the converses. The converse of Lemma 28(even for n^2 replaced by some polynomial) is false. For example, consider the very weak Pigeonhole Principle of [15].

References

1 Michael Alekhnovich. Lower Bounds for k-DNF Resolution on Random 3-CNFs. *Computational Complexity*, 20(4):597–614, 2011. doi:10.1007/s00037-011-0026-0.

2 Razborov Alexander A. Pseudorandom generators hard for k-DNF resolution and polynomial calculus resolution. *Annals of Mathematics*, 181(41):415–472, 2015.

3 Albert Atserias. Improved bounds on the Weak Pigeonhole Principle and infinitely many primes from weaker axioms. *Theor. Comput. Sci.*, 295:27–39, 2003. doi:10.1016/S0304-3975(02)00394-8.

4 Albert Atserias, Ilario Bonacina, Susanna F. de Rezende, Massimo Lauria, Jakob Nordström, and Alexander A. Razborov. Clique is hard on average for regular resolution. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 866–877. ACM, 2018. doi:10.1145/3188745.3188856.

5 Albert Atserias, Maria Luisa Bonet, and Juan Luis Esteban. Lower Bounds for the Weak Pigeonhole Principle and Random Formulas beyond Resolution. *Inf. Comput.*, 176(2):136–152, 2002. doi:10.1006/inco.2002.3114.

6 Paul Beame, Russell Impagliazzo, and Ashish Sabharwal. Resolution Complexity of Independent Sets in Random Graphs. In *Proceedings of the 16th Annual IEEE Conference on Computational Complexity, Chicago, Illinois, USA, June 18-21, 2001*, pages 52–68. IEEE Computer Society, 2001. doi:10.1109/CCC.2001.933872.

7 Paul Beame and Toniann Pitassi. Simplified and Improved Resolution Lower Bounds. In *37th Annual Symposium on Foundations of Computer Science, FOCS '96, Burlington, Vermont, USA, 14-16 October, 1996*, pages 274–282. IEEE Computer Society, 1996. doi:10.1109/SFCS.1996.548486.

8 Eli Ben-sasson and Avi Wigderson. Short proofs are narrow - resolution made simple. In *Journal of the ACM*, pages 517–526, 1999.

9 Olaf Beyersdorff, Nicola Galesi, and Massimo Lauria. A lower bound for the pigeonhole principle in tree-like Resolution by asymmetric Prover-Delayer games. *Inf. Process. Lett.*, 110(23):1074–1077, 2010. doi:10.1016/j.ipl.2010.09.007.

10 Olaf Beyersdorff, Nicola Galesi, and Massimo Lauria. Parameterized Complexity of DPLL Search Procedures. *ACM Trans. Comput. Logic*, 14(3):20:1–20:21, August 2013. doi:10.1145/2499937.2499941.

11 Olaf Beyersdorff, Nicola Galesi, Massimo Lauria, and Alexander A. Razborov. Parameterized Bounded-Depth Frege Is not Optimal. *TOCT*, 4(3):7:1–7:16, 2012. doi:10.1145/2355580.2355582.

12 Ilario Bonacina and Nicola Galesi. A Framework for Space Complexity in Algebraic Proof Systems. *J. ACM*, 62(3):23:1–23:20, 2015. doi:10.1145/2699438.

13 Ilario Bonacina, Nicola Galesi, and Neil Thapen. Total Space in Resolution. *SIAM J. Comput.*, 45(5):1894–1909, 2016. doi:10.1137/15M1023269.

14 Maria Luisa Bonet and Nicola Galesi. Optimality of size-width tradeoffs for resolution. *Computational Complexity*, 10(4):261–276, 2001. doi:10.1007/s000370100000.

15 Samuel R. Buss and Toniann Pitassi. Resolution and the Weak Pigeonhole Principle. In *Computer Science Logic, 11th International Workshop, CSL '97, Annual Conference of the EACSL, Aarhus, Denmark, August 23-29, 1997, Selected Papers*, pages 149–156, 1997. doi:10.1007/BFb0028012.

16 Stefan S. Dantchev and Søren Riis. Tree Resolution Proofs of the Weak Pigeon-Hole Principle. In *Proceedings of the 16th Annual IEEE Conference on Computational Complexity, Chicago, Illinois, USA, June 18-21, 2001*, pages 69–75, 2001. doi:10.1109/CCC.2001.933873.

17 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.

18 Juan Luis Esteban, Nicola Galesi, and Jochen Messner. On the complexity of resolution with bounded conjunctions. *Theor. Comput. Sci.*, 321(2-3):347–370, 2004. doi:10.1016/j.tcs.2004.04.004.

19 Yuval Filmus, Massimo Lauria, Jakob Nordström, Noga Ron-Zewi, and Neil Thapen. Space Complexity in Polynomial Calculus. *SIAM J. Comput.*, 44(4):1119–1153, 2015. doi:10.1137/120895950.

20 Nicola Galesi and Massimo Lauria. Optimality of size-degree tradeoffs for polynomial calculus. *ACM Trans. Comput. Log.*, 12(1):4:1–4:22, 2010. doi:10.1145/1838552.1838556.

21 Armin Haken. The Intractability of Resolution. *Theor. Comput. Sci.*, 39:297–308, 1985.

22 Pavel Hrubes and Pavel Pudlák. Random Formulas, Monotone Circuits, and Interpolation. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 121–131. IEEE Computer Society, 2017. doi:10.1109/FOCS.2017.20.

23 Jan Krajíček. *Bounded arithmetic, propositional logic and complexity theory*. Cambridge University Press, 1995.

24 Balakrishnan Krishnamurthy. Short Proofs for Tricky Formulas. *Acta Inf.*, 22(3):253–275, 1985. doi:10.1007/BF00265682.

25 Oliver Kullmann. Investigating a general hierarchy of polynomially decidable classes of CNF's based on short tree-like resolution proofs. *Electronic Colloquium on Computational Complexity (ECCC)*, 41, 1999. URL: http://eccc.hpi-web.de/eccc-reports/1999/TR99-041/index.html.

26 G. Kwon and W. Klieber. Efficient CNF Encoding for Selecting 1 from N Objects. In *Fourth Workshop on Constraints in Formal Verification (CFV '07)*, 2007.

27 Massimo Lauria, Pavel Pudlák, Vojtech Rödl, and Neil Thapen. The complexity of proving that a graph is Ramsey. *Combinatorica*, 37(2):253–268, 2017. doi:10.1007/s00493-015-3193-9.

28 Alexis Maciel, Toniann Pitassi, and Alan R. Woods. A New Proof of the Weak Pigeonhole Principle. *J. Comput. Syst. Sci.*, 64(4):843–872, 2002. doi:10.1006/jcss.2002.1830.

29 Justyna Petke. *Bridging Constraint Satisfaction and Boolean Satisfiability*. Artificial Intelligence: Foundations, Theory, and Algorithms. Springer, 2015. doi:10.1007/978-3-319-21810-6.

30 P. Pudlák. Proofs as games. *American Mathematical Monthly*, pages 541–550, June-July 2000.

31 Ran Raz. Resolution lower bounds for the weak pigeonhole principle. *J. ACM*, 51(2):115–138, 2004. doi:10.1145/972639.972640.

32 Alexander A. Razborov. Proof Complexity of Pigeonhole Principles. In Werner Kuich, Grzegorz Rozenberg, and Arto Salomaa, editors, *Developments in Language Theory*, pages 100–116, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg.

33 Alexander A. Razborov. Resolution lower bounds for the weak functional pigeonhole principle. *Theor. Comput. Sci.*, 1(303):233–243, 2003. doi:10.1016/S0304-3975(02)00453-X.

34 Søren Riis. A complexity gap for tree resolution. *Computational Complexity*, 10(3):179–209, 2001. URL: http://link.springer.de/link/service/journals/00037/bibs/1010003/10100179.htm.

35 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.

36 Gunnar Stålmarck. Short Resolution Proofs for a Sequence of Tricky Formulas. *Acta Inf.*, 33(3):277–280, 1996. doi:10.1007/s002360050044.

37 Toby Walsh. SAT v CSP. In *Principles and Practice of Constraint Programming - CP 2000, 6th International Conference, Singapore, September 18-21, 2000, Proceedings*, pages 441–456, 2000. doi:10.1007/3-540-45349-0_32.

Fourier Bounds and Pseudorandom Generators for Product Tests

Chin Ho Lee
Northeastern University, Boston, USA
chlee@ccs.neu.edu

Abstract

We study the Fourier spectrum of functions $f\colon \{0,1\}^{mk} \to \{-1,0,1\}$ which can be written as a product of k Boolean functions f_i on disjoint m-bit inputs. We prove that for every positive integer d,

$$\sum_{S\subseteq[mk]:|S|=d} |\hat{f_S}| = O\big(\min\{m, \sqrt{m\log(2k)}\}\big)^d.$$

Our upper bounds are tight up to a constant factor in the $O(\cdot)$. Our proof uses Schur-convexity, and builds on a new "level-d inequality" that bounds above $\sum_{|S|=d} \hat{f_S}^2$ for any $[0,1]$-valued function f in terms of its expectation, which may be of independent interest.

As a result, we construct pseudorandom generators for such functions with seed length $\tilde{O}(m + \log(k/\varepsilon))$, which is optimal up to polynomial factors in $\log m$, $\log\log k$ and $\log\log(1/\varepsilon)$. Our generator in particular works for the well-studied class of combinatorial rectangles, where in addition we allow the bits to be read in any order. Even for this special case, previous generators have an extra $\tilde{O}(\log(1/\varepsilon))$ factor in their seed lengths.

We also extend our results to functions f_i whose range is $[-1,1]$.

2012 ACM Subject Classification Theory of computation → Pseudorandomness and derandomization

Keywords and phrases bounded independence plus noise, Fourier spectrum, product test, pseudorandom generators

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.7

Funding Supported by NSF CCF award 1813930. Work done in part while visiting Amnon Ta-Shma at Tel Aviv University, with support from the Blavatnik family fund and ISF grant no. 952/18.

Acknowledgements I thank Salil Vadhan for asking about the Fourier spectrum of product tests. I also thank Andrej Bogdanov, Gil Cohen, Amnon Ta-Shma, Avishay Tal and Emanuele Viola for very helpful conversations. I am grateful to Emanuele Viola for his invaluable comments on the write-up, and Dean Doron, Pooya Hatami and William Hoza for pointing out a simplification and improvement of Theorem 8. Finally, I thank the anonymous reviewers for their useful comments.

1 Introduction

In this paper we study tests on n bits which can be written as a product of k bounded real-valued functions defined on disjoint inputs of m bits. We first define them formally.

▶ **Definition 1** (Product tests). *A function $f\colon \{0,1\}^n \to [-1,1]$ is a* product test *with k functions of input length m if there exist k disjoint subsets $I_1, I_2, \ldots, I_k \subseteq \{1,2,\ldots,n\}$ of size $\leq m$ such that $f(x) = \prod_{i\leq k} f_i(x_{I_i})$ for some functions f_i with range in $[-1,1]$. Here x_{I_i} are the $|I_i|$ bits of x indexed by I_i.*

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka; Article No. 7; pp. 7:1–7:25

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


More generally, the range of each function f_i can be $\mathbb{C}_{\leq 1} := \{z \in \mathbb{C} : |z| \leq 1\}$, the complex unit disk [22, 26], or the set of square matrices over a field [44]. However, in this paper we only focus on the range $[-1, 1]$. As we will soon explain, our results do not hold for the broader range of $\mathbb{C}_{\leq 1}$.

The class of product tests was first introduced by Gopalan, Kane and Meka under the name of *Fourier shapes* [22]. However, in their definition, the subsets I_i are fixed. Motivated by the recent constructions of pseudorandom generators against *unordered* tests, which are tests that read input bits in arbitrary order [8, 28, 44, 50], Haramaty, Lee and Viola [26] considered the generalization in which the subsets I_i can be arbitrary as long as they are of bounded size and pairwise disjoint.

Product tests generalize several restricted classes of tests. For example, when the range of the functions f_i is $\{0, 1\}$, product tests correspond to the AND of disjoint Boolean functions, also known as the well-studied class of *combinatorial rectangles* [4, 40, 41, 30, 20, 7, 36, 56, 23, 25]. When the range of the f_i is $\{-1, 1\}$, they correspond to the XOR of disjoint Boolean functions, also known as the class of *combinatorial checkerboards* [57]. More importantly, product tests also capture *read-once space computation*. Specifically, Reingold, Steinke and Vadhan [44] showed that the class of read-once width-w branching programs can be encoded as product tests with outputs $\{0, 1\}^{w \times w}$, the set of $w \times w$ Boolean matrices.

In the past year, the study of product tests [26, 33] has found applications in constructing state-of-the-art pseudorandom generators (PRGs) for space-bounded algorithms. Using ideas in [23, 25, 33, 14], Meka, Reingold and Tal [38] constructed a pseudorandom generator for width-3 read-once branching programs (ROBPs) on n bits with seed length $\tilde{O}(\log n \log(1/\varepsilon))$, giving the first improvement of Nisan's generator [40] in the 90s. Building on [44, 26, 14], Forbes and Kelley significantly simplified the analysis of [38] and constructed a generator that fools *unordered* polynomial-width read-once branching programs. Thus, it is motivating to further study product tests, in the hope of gaining more insights into constructing better generators for space-bounded algorithms, and resolving the long-standing open problem of RL vs. L.

In this paper we are interested in understanding the Fourier spectrum of product tests. We first define the *Fourier weight* of a function. For a function $f\colon \{0, 1\}^n \to \mathbb{R}$, consider its Fourier expansion $f = \sum_{S \subseteq [n]} \hat{f}_S \chi_S$.

▶ **Definition 2** (dth level Fourier weight in L_q-norm)**.** *Let $f\colon \{0, 1\}^n \to \mathbb{C}_{\leq 1}$ be any function. The d*th level Fourier weight of f in L_q-norm *is*

$$W_{q,d}[f] := \sum_{|S|=d} |\hat{f}_S|^q.$$

We denote by $W_{q,\leq d}[f]$ the sum $\sum_{\ell=0}^{d} W_{q,\ell}[f]$.

Several papers have studied the Fourier spectrum of different classes of tests. This includes constant-depth circuits [37, 51], read-once branching programs [44, 50, 14], and low-sensitivity functions [24]. More specifically, these papers showed that they have *bounded L_1 Fourier tail*, that is, there exists a positive number b such that for every test f in the class and every positive integer d, we have

$$W_{1,d}[f] \leq b^d.$$

One technical contribution of this paper is giving tight upper and lower bounds on the L_1 Fourier tail of product tests.

▶ **Theorem 3.** *Let $f\colon \{0,1\}^n \to [-1,1]$ be a product test of k functions $f_1, \ldots, f_k$ with input length m. Suppose there is a constant $c > 0$ such that $|\mathbb{E}[f_i]| \leq 1 - 2^{-cm}$ for every f_i. For every positive integer d, we have*

$$W_{1,d}[f] \leq \left(72(\sqrt{c} \cdot m)\right)^d.$$

Theorem 3 applies to Boolean functions f_i with outputs $\{0,1\}$ or $\{-1,1\}$, for which we know a bound on c. Moreover, the parity function on mk bits can be written as a product test with outputs $\{-1,1\}$, which has $\hat{f}_{[mk]} = 1$. So product tests do not have non-trivial L_2 Fourier tail. (See [51] for a definition.)

We also obtain a different upper bound when the f_i are arbitrary $[-1,1]$-valued functions.

▶ **Theorem 4.** *Let $f\colon \{0,1\}^n \to [-1,1]$ be a product test of k functions $f_1, \ldots, f_k$ with input length m. Let d be a positive integer. We have*

$$W_{1,d}[f] \leq \left(85\sqrt{m \ln(4ek)}\right)^d.$$

We note that Theorems 3 and 4 are incomparable, as one can take $m = 1$ and $k = n$, or $m = n$ and $k = 1$.

▷ Claim 5. For all positive integers m and d, there exists a product test $f\colon \{0,1\}^{mk} \to \{0,1\}$ with $k = d \cdot 2^m$ functions of input length m such that

$$W_{1,d}[f] \geq (m/e^{3/2})^d.$$

This matches the upper bound $W_{1,d}[f] = O(m)^d$ in Theorem 3 up to the constant in the $O(\cdot)$. Moreover, applying Theorem 4 to the product test f in Claim 5 gives $W_{1,d}[f] = O(\sqrt{m \log(2k)})^d = O(m + \sqrt{m \log d})^d$. Therefore, for all integers m and $d \leq 2^{O(m)}$, there exists an integer k and a product test f such that the upper bound $W_{1,d}[f] = O(\sqrt{m \log(2k)})^d$ is tight up to the constant in the $O(\cdot)$.

We now discuss some applications of Theorems 3 and 4 in pseudorandomness.

Pseudorandom generators

In recent years, researchers have developed new frameworks to construct pseudorandom generators against different classes of tests. Gopalan, Meka, Reingold, Trevisan and Vadhan [23] refined a framework introduced by Ajtai and Wigderson [5] to construct better generators for the classes of combinatorial rectangles and read-once DNFs. Since then, this framework has been used extensively to construct new PRGs against different classes of tests [53, 22, 25, 44, 50, 15, 26, 27, 46, 33, 14, 21, 38, 19]. Recently, a beautiful work by Chattopadhyay, Hatami, Hosseini and Lovett [12] developed a new framework of constructing PRGs against any classes of functions that are closed under restriction and have bounded L_1 Fourier tail. Thus, applying their result to Theorems 3 and 4, we can immediately obtain a non-trivial PRG for product tests. However, using the recent result of Forbes and Kelley [21] and exploiting the structure of product tests, we use the Ajtai–Wigderson framework to construct PRGs with much better seed length than using [12] as a blackbox.

▶ **Theorem 6.** *There exists an explicit generator $G\colon \{0,1\}^\ell \to \{0,1\}^n$ that fools the XOR of any k Boolean functions on disjoint inputs of length $\leq m$ with error ε and seed length $O(m + \log(n/\varepsilon))(\log m + \log\log(n/\varepsilon))^2 = \tilde{O}(m + \log(n/\varepsilon))$.*

Here $\tilde{O}(1)$ hides polynomial factors in $\log m$, $\log\log k$, $\log\log n$ and $\log\log(1/\varepsilon)$. When $mk = n$ or $\varepsilon = n^{-\Omega(1)}$, the generator in Theorem 6 has seed length $\tilde{O}(m + \log(k/\varepsilon))$, which is optimal up to $\tilde{O}(1)$ factors.

We now compare Theorem 6 with previous works. Using a completely different analysis, Lee and Viola [33] obtained a generator with seed length $\tilde{O}((m + \log k))\log(1/\varepsilon)$. When $m = O(\log n)$ and $k = 1/\varepsilon = n^{\Omega(1)}$, this is $\tilde{O}(\log^2 n)$, whereas the generator in Theorem 6 has seed length $\tilde{O}(\log n)$. When each function f_i is computable by a read-once width-w branching program on m bits, Meka, Reingold and Tal [38] obtained a PRG with seed length $O(\log(n/\varepsilon))(\log m + \log\log(n/\varepsilon))^{2w+2}$. When $m = O(\log(n/\varepsilon))$, Theorem 6 improves on their generator on the lower order terms. As a result, we obtain a PRG for *read-once* $\mathbb{F}_2$*-polynomials*, which are a sum of monomials on disjoint variables over $\mathbb{F}_2$, with seed length $O(\log n/\varepsilon)(\log\log(n/\varepsilon))^2$. This also improves on the seed length of their PRG for read-once polynomials in the lower order terms by a factor of $(\log\log(n/\varepsilon))^4$.

Our generator in Theorem 6 also works for the AND of the functions f_i, corresponding to the class of *unordered* combinatorial rectangles. Previous generators [11, 17] use almost-bounded independence or small-bias distributions, and have seed length $O(\log(n/\varepsilon))(1/\varepsilon)$. While several papers [36, 56, 23, 25, 22] have improved the seed length for this model in the *fixed* order setting, our generator is the first improvement for the *unordered* setting and has nearly-optimal seed length. In fact, we have the following more general corollary.

▶ **Corollary 7.** *There exists an explicit pseudorandom generator $G\colon \{0,1\}^\ell \to \{0,1\}^n$ with seed length $\tilde{O}(m+\log(n/\varepsilon))$ such that the following holds. Let $f_1, \dots, f_k\colon \{0,1\}^{I_i} \to \{0,1\}$ be k Boolean functions where the subsets $I_i \subseteq [n]$ are pairwise disjoint and have size at most m. Let $g\colon \{0,1\}^k \to \mathbb{C}_{\le 1}$ be any function and write g in its Fourier expansion $g = \sum_{S\subseteq[k]} \hat{g}_S\chi_S$. Then G fools $g(f_1, \dots, f_k)$ with error $L_1[g]\cdot\varepsilon$, where $L_1[g] := \sum_{S\neq\emptyset}|\hat{g}_S|$.*

Proof. Let G be the generator in Theorem 6. Note that $\chi_S(f_1(x_{I_1}), \dots, f_k(x_{I_k}))$ is a product test with outputs $\{-1, 1\}$. So by Theorem 6 we have

$$\begin{aligned}
&\left|\mathbb{E}[g(f_1(U_{I_1}), \dots, f_k(U_{I_k})) - \mathbb{E}[g(f_1(G_{I_1}), \dots, f_k(G_{I_k})]\right| \\
&\le \sum_S |\hat{g}_S| \left|\mathbb{E}[\chi_S(f_1(U_{I_1}), \dots, f_k(U_{I_k}))] - \mathbb{E}[\chi_S(f_1(G_{I_1}), \dots, f_k(G_{I_k})]\right| \\
&\le L_1[g]\cdot\varepsilon.
\end{aligned}$$

◀

Note that the AND function has $L_1[\text{AND}] \le 1$, and so the generator in Corollary 7 fools unordered combinatorial rectangles.

When the functions f_i in the product tests have outputs $[-1, 1]$, we also obtain the following generator.

▶ **Theorem 8.** *There exists an explicit generator $G\colon \{0,1\}^\ell \to \{0,1\}^n$ that fools any product test with k functions of input length m with error ε and seed length $O(\log mk)((m + \log(k/\varepsilon))(\log m + \log\log(k/\varepsilon)) + \log\log n) = \tilde{O}(m + \log(k/\varepsilon))\log k$.*

When $m = o(\log n)$ and $k = 1/\varepsilon = 2^{o(\sqrt{\log n})}$, Theorem 8 gives a better seed length than Theorem 6. Thus the generator in Theorem 8 remains interesting for $f_i \in \{-1, 1\}$ when a product test f depends on very few variables and the error ε is not so small.

Previous best generator [33] has an extra $\tilde{O}(\log(1/\varepsilon))$ in the seed length. However, the generator in [33] works even when the f_i have range $\mathbb{C}_{\le 1}$, which implies generators for several variants of product tests such as generalized halfspaces and combinatorial shapes. (See [22] for the reductions.)

Finally, when the subsets I_i of a product test are fixed and known in advanced, Gopalan, Kane and Meka [22] constructed a PRG of the same seed length as Theorem 6, but again their PRG works more generally for the range of $\mathbb{C}_{\leq 1}$ instead of $\{-1, 1\}$.

$\mathbb{F}_2$-polynomials

Chattopadhyay, Hatami, Lovett and Tal [13] recently constructed a pseudorandom generator for any class of functions that are closed under restriction, provided there is an upper bound on the second level Fourier weight of the functions in L_1-norm. They conjectured that every n-variate $\mathbb{F}_2$-polynomial f of degree d satisfies the bound $W_{1,2}[f] = O(d^2)$. In particular, a bound of $n^{1/2-o(1)}$ would already imply a generator for polynomials of degree $d = \Omega(\log n)$, a major breakthrough in complexity theory. Theorem 4 shows that their conjecture is true for the special case of *read-once* polynomials. In fact, it shows that $W_{1,t}[f] = O(d^t)$ for every positive integer t. Previous bound for read-once polynomials gives $W_{1,t}[f] = O(\log^4 n)^t$ [14].

The coin problem

Let $X_{n,\varepsilon} = (X_1, \ldots, X_n)$ be the distribution over n bits, where the variables X_i are independent and each X_i equals 1 with probability $(1-\varepsilon)/2$ and 0 otherwise. The *ε-coin problem* asks whether a given function f can distinguish between the distributions $X_{n,\varepsilon}$ and $X_{n,0}$ with advantage $1/3$.

This central problem has wide range of applications in computational complexity and has been studied extensively for different restricted classes of tests, including bounded-depth circuits [2, 54, 3, 6, 55, 47, 1, 56, 16], space-bounded algorithms [9, 49, 16], bounded-depth circuits with parity gates [47, 32, 45, 35], $\mathbb{F}_2$-polynomials [35, 13] and product tests [34].

It is known that if a function f has bounded L_1 Fourier tail, then it implies a lower bound on the smallest ε^* of ε that f can solve the ε-coin problem.

▶ **Fact 9.** *Let $f\colon \{0,1\}^n \to \mathbb{C}_{\leq 1}$ be any function. If for every integer $d \in \{0, \ldots, n\}$ we have $W_{1,d}[f] \leq b^d$, then f solves the ε-coin problem with advantage at most $2b\varepsilon$.*

Proof. We may assume $b\varepsilon \leq 1/2$, otherwise the result is trivial. Observe that we have $\mathbb{E}[\chi_S(X_{n,\varepsilon})] = \varepsilon^{|S|}$ for every subset $S \subseteq [n]$. Thus,

$$\begin{aligned}\left|\mathbb{E}[f(X_{n,\varepsilon})] - \mathbb{E}[f(X_{n,0})]\right| &= \left|\sum_{S\neq\emptyset} \hat{f}_S \,\mathbb{E}[X_{n,\varepsilon}]\right| \\ &\leq \sum_{d=1}^{n} \sum_{|S|=d} |\hat{f}_S| \cdot \varepsilon^d = \sum_{d=1}^{n} (b\varepsilon)^d \leq b\varepsilon \cdot \sum_{d=1}^{n} 2^{-(d-1)} \leq 2b\varepsilon. \qquad \blacktriangleleft\end{aligned}$$

Lee and Viola [34] showed that product tests with range $[-1, 1]$ can solve the ε-coin problem with $\varepsilon^* = \Theta(1/\sqrt{m \log k})$. Hence, Fact 9 implies that Theorem 4 recovers their lower bound. Moreover, their upper bound implies that the dependence on m and k in Theorem 4 is tight up to constant factors when d is constant. Claim 5 complements this by showing that the dependence on d in Theorem 4 is also tight for some choice of k.

The work [34] also shows that when the range of the functions f_i is $\mathbb{C}_{\leq 1}$, the right answer for ε^* is $\Theta(1/\sqrt{mk})$. Therefore, one cannot hope for a better tail bound than the trivial bound of $(\sqrt{mk})^d$ when the range is $\mathbb{C}_{\leq 1}$.

1.1 Techniques

We now explain how to obtain Theorems 3 and 4 and our pseudorandom generators for product tests (Theorems 6 and 8).

1.1.1 Fourier spectrum of product tests

The high-level idea of proving Theorems 3 and 4 is inspired from [34]. For intuition, let us first assume that the functions f_i have outputs $\{0,1\}$ and are all equal to f_1 (but defined on disjoint inputs). It will also be useful to think of the number of functions k being much larger than input length m of each function. We first explain how to bound above $W_{1,1}[f]$. (Recall in Definition 2 we defined $W_{q,d}[f]$ of a function f to be $\sum_{|S|=d}|\hat{f}_S|^q$.)

Bounding $W_{1,1}[f]$

Since the functions f_i of a product test f are defined on disjoint inputs, each Fourier coefficient of f is a product of the coefficients of the f_i, and so each weight-1 coefficent of f is a product of $k-1$ weight-0 and 1 weight-1 coefficients of the f_i. From this, we can see that $W_{1,1}[f]$ is equal to

$$\binom{k}{1} \cdot W_{1,1}[f_1] \cdot W_{1,0}[f_1]^{k-1} = k \cdot W_{1,1}[f_1] \cdot \mathbb{E}[f_1]^{k-1}. \tag{1}$$

Because of the term $\mathbb{E}[f_1]^{k-1}$, to maximize $W_{1,1}[f]$ it is natural to consider taking f_1 to be a function with expectation $\mathbb{E}[f_1]$ as close to 1 as possible, i.e. the OR function. In such case, one would hope for a better bound on $W_{1,1}[f_1]$. Indeed, Chang's inequality [10] (see also [29] for a simple proof) says that for a $[0,1]$-valued function g with expectation $\alpha \leq 1/2$, we have

$$W_{2,1}[g] \leq 2\alpha^2 \ln(1/\alpha).$$

(The condition $\alpha \leq 1/2$ is without loss of generality as one can instead consider $1-g$.) It follows by a simple application of the Cauchy–Schwarz inequality that $W_{1,1}[g] \leq O(\sqrt{n}) \cdot \alpha\sqrt{\ln(1/\alpha)}$ (see Fact 12 below for a proof). Moreover, when the functions f_i are Boolean, we have $2^{-m} \leq \mathbb{E}[f_i] \leq 1 - 2^{-m}$, and so $\sqrt{\ln(1/\alpha)} \leq \sqrt{m}$. Plugging these bounds into Equation (1), we obtain a bound of $O(m) \cdot k(1-\mathbb{E}[f_1])\,\mathbb{E}[f_1]^{k-1}$. So indeed $\mathbb{E}[f_1]$ should be roughly $1-1/k$ in order to maximize $W_{1,1}[f]$, giving an upper bound of $O(m)$. For the case where the f_i can be different, a simple convexity argument shows that $W_{1,1}[f]$ is maximized when the functions f_i have the same expectation.

Bounding $W_{1,d}[f]$ for $d > 1$

To extend this argument to $d > 1$, one has to generalize Chang's inequality to bound above $W_{2,d}[g]$ for $d > 1$. The case $d = 2$ was already proved by Talagrand [52]. Following Talagrand's argument in [52] and inspired by the work of Keller and Kindler [31], which proved a similar bound in terms of a different measure than $\mathbb{E}[g]$, we prove the following bound on $W_{2,d}[g]$ in terms of its expectation.

▶ **Lemma 10.** *Let $g\colon \{0,1\}^n \to [0,1]$ be any function. For every positive integer d, we have*

$$W_{2,d}[g] \leq 4\,\mathbb{E}[g]^2\big(2e\ln(e/\,\mathbb{E}[g]^{1/d})\big)^d.$$

We note that the exponent $1/d$ of $\mathbb{E}[g]$ either did not appear in previous upper bounds (mentioned without proof in [29]), or only holds for restricted values of d [42]. This exponent is not important for proving Theorem 3 , but will be crucial in the proof of Theorem 4, which we will explain later on.

For $d > 1$, the expression for $W_{1,d}[f]$ becomes much more complicated than $W_{1,1}[f]$, as it involves $W_{1,z}[f_1]$ for different values of $z \in [m]$. So one has to formulate the expression of $W_{1,d}[f]$ carefully (see Lemma 13). Once we have obtained the right expression for $W_{1,d}[f]$,

the proof of Theorem 3 follows the outline above by replacing Chang's inequality with Lemma 10. One can then handle functions f_i with outputs $\{-1,1\}$ by considering the translation $f_i \mapsto (1-f_i)/2$, which only changes each $W_{1,d}[f_i]$ (for $d>0$) by a factor of 2. We remark that Theorem 3 is sufficient for constructing the generator in Theorem 6.

Handling $[-1,1]$-valued f_i

Extending this argument to proving Theorem 4 poses several challenges. Following the outline above, after plugging in Lemma 10, we would like to show that $\mathbb{E}[f_1]$ should be roughly $1-1/k$ to maximize $W_{1,d}[f]$. However, it is no longer clear why this is the case even assuming the maximum is attained by functions f_i with the same expectation, as we now do not have the bound $\sqrt{\ln(1/\alpha)} \le \sqrt{m}$, and so it cannot be used to simplify the expression of $W_{1,d}[f]$ as before. In fact, the above assumption is simply false if we plug in the upper bound in Lemma 10 with the exponent $1/d$ omitted to the $W_{1,z_i}[f_i]$.

Using Lemma 10 and the symmetry of the expression for $W_{1,d}[f]$, we reduce the problem of bounding above $W_{1,d}[f]$ with different f_i to bounding the same quantity but with the additional assumption that the f_i have the same expectation $\mathbb{E}[f_1]$. This uses Schur-convexity (see Section 2 for its definition). Then by another convexity argument we show that the maximum is attained when $\mathbb{E}[f_1]$ is roughly equal to $1-d/k$. Both of these arguments critically rely on the aforementioned exponent of $1/d$ in Lemma 10.

1.1.2 Pseudorandom generators

We now discuss how to use Theorems 3 and 4 to construct our pseudorandom generators for product tests. Our construction follows the Ajtai–Wigderson framework [5] that was recently revived and refined by Gopalan, Meka, Reingold, Trevisan and Vadhan [23].

The high-level idea of this framework involves two steps. For the first step, we show that *derandomized bounded independence plus noise* fools f. More precisely, we will show that if we start with a small-bias or almost-bounded independent distribution D ("bounded independence"), and select roughly half of D's positions T pseudorandomly and set them to uniform U ("plus noise"), then this distribution, denoted by $D+T\wedge U$, fools product tests.

Forbes and Kelley [21] recently improved the analysis in [26] and implicitly showed that δ-almost d-wise independent plus noise fools product tests, where $d=O(m+\log(k/\varepsilon))$ and $\delta = n^{-\Omega(d)}$. Using Theorem 4, we improve the dependence on δ to $(m\ln k)^{-\Omega(d)}$ and obtain the following theorem.

▶ **Theorem 11.** *Let $f\colon \{0,1\}^n \to [-1,1]$ be a product test with k functions of input length m. Let d be a positive integer. Let D and T be two independent δ-almost d-wise independent distributions over $\{0,1\}^n$, and U be the uniform distribution over $\{0,1\}^n$. Then*

$$\left|\mathbb{E}[f(D+T\wedge U)] - \mathbb{E}[f(U)]\right| \le k\cdot\left(\sqrt{\delta}\cdot(170\cdot\sqrt{m\ln(ek)})^d + 2^{-(d-m)/2}\right),$$

where "$+$" and "$\wedge$" are bit-wise XOR and AND respectively.

The second step of the Ajtai–Wigderson framework builds a pseudorandom generator by applying the first step (Theorem 11) recursively. Let $f\colon \{0,1\}^n \to \{0,1\}$ be a product test with k functions of input length m. As product tests are closed under restrictions (and shifts), after applying Theorem 11 to f and fixing D and T in the theorem, the function $f_{D,T}\colon \{0,1\}^T \to \{0,1\}$ defined by $f_{D,T}(y) := f(D+T\wedge y)$ is also a product test. Thus one can apply Theorem 11 to $f_{D,T}$ again and repeat the argument recursively. We will use different progress measures to bound above the number of recursion steps in our constructions. We first describe the recursion in Theorem 8 as it is simpler.

Fooling $[-1, 1]$-valued product tests

Here our progress measure is the number of bits that are defined by the product test f. We show that after $O(\log(mk))$ steps of the recursion, the restricted product test is defined on at most $O(m + \log(k/\varepsilon))$ bits with high probability, which can then be fooled by an almost-bounded independent distribution. This simple recursion gives our second PRG (Theorem 8).

Fooling Boolean-valued product tests

Our construction of the first generator (Theorem 6) is more complicated and uses two progress measures. The first one is the maximum input length m of the functions f_i, and the second is the number k of the functions f_i. We reduce the number of recursion steps from $O(\log(k/\varepsilon)) \log m$ to $O(\log m)$. This requires a more delicate construction and analysis that are similar to the recent work of Meka, Reingold and Tal [38], which constructed a pseudorandom generator against XOR of disjoint constant-width read-once branching programs. There are two main ideas in their construction. First, they ensure $k \leq 16^m$ in each step of the recursion, by constructing another PRG to fool the test f for the case $k \geq 16^m$. We will also use this PRG in our construction. Next, throughout the recursion they allow one "bad" function f_i of the product test f to have a longer input length than m, but not longer than $O(\log(n/\varepsilon))$. Using these two ideas, they show that whenever $m \geq \log \log n$ during the recursion, then after $O(1)$ steps of the recursion all but the "bad" f_i have their input length restricted by a half, while the "bad" f_i always has length $O(\log(n/\varepsilon))$. This allows us to repeat $O(\log m)$ steps until we are left with a product test of $k' \leq \text{polylog}(n)$ functions, where all but one of the f_i have input length at most $m' = O(\log \log n)$.

Now we switch our progress measure to the number of functions. This part is different from [38], in which their construction relies on the fact that the f_i are computable by read-once branching programs. Here because our functions f_i are arbitrary, by grouping c functions as one, we can instead think of the parameters k' and m' in the product test as $k'' = k'/c$ and $m'' = cm'$, respectively. Choosing c to be $O(\log n / \log \log n)$, we have $m'' = O(\log n)$ and so we can repeat the previous argument again. Because each time k' is reduced by a factor of c, after repeating this for $O(1)$ steps, we are left with a product test defined on $O(\log n)$ bits, which can be fooled using a small-bias distribution. This gives our first generator (Theorem 6).

Organization

In Section 2 we prove Theorems 3 and 4. In Section 3 we construct our pseudorandom generators for product tests, proving Theorems 6 and 8. In Section 4 we prove Lemma 10, which is used in the proof of Theorem 4.

2 Fourier spectrum of product tests

In this section we prove Theorems 3 and 4. We first restate the theorems.

▶ **Theorem 3.** *Let $f\colon \{0,1\}^n \to [-1,1]$ be a product test of k functions $f_1, \ldots, f_k$ with input length m. Suppose there is a constant $c > 0$ such that $|\mathbb{E}[f_i]| \leq 1 - 2^{-cm}$ for every f_i. For every positive integer d, we have*

$$W_{1,d}[f] \leq \left(72(\sqrt{c} \cdot m)\right)^d.$$

▶ **Theorem 4.** *Let $f\colon \{0,1\}^n \to [-1,1]$ be a product test of k functions $f_1,\dots,f_k$ with input length m. Let d be a positive integer. We have*

$$W_{1,d}[f] \le \big(85\sqrt{m\ln(4ek)}\big)^d.$$

Both theorems rely on the following lemma which gives an upper bound on $W_{2,d}[g]$ in terms of the expectation of a $[0,1]$-valued function g. The case $d=1$ is known as Chang's inequality [10]. (See also [29] for a simple proof.) This was then generalized by Talagrand to $d=2$ [52]. Using a similar argument to [52], we extend this to $d>2$.

▶ **Lemma 10.** *Let $g\colon \{0,1\}^n \to [0,1]$ be any function. For every positive integer d, we have*

$$W_{2,d}[g] \le 4\,\mathbb{E}[g]^2\big(2e\ln(e/\mathbb{E}[g]^{1/d})\big)^d.$$

We defer its proof to Section 4. We remark that a similar upper bound was proved by Keller and Kindler [31]. However, the upper bound in [31] was proved in terms of $\sum_{i=1}^n I_i[g]^2$, where $I_i[g]$ is the influence of the ith coordinate on g, instead of $\mathbb{E}[g]$. A similar upper bound in terms of $\mathbb{E}[g]$ can be found in [42] under the extra condition $d \le 2\ln(1/\mathbb{E}[g])$.

We will also use the following well-known fact that bounds above $W_{1,d}[f]$ in terms of $W_{2,d}[f]$.

▶ **Fact 12.** *Let $f\colon \{0,1\}^n \to \mathbb{R}$ be any function. We have $W_{1,d}[f] \le n^{d/2}\sqrt{W_{2,d}[f]}$.*

Proof. By the Cauchy–Schwarz inequality,

$$W_{1,d}[f] = \sum_{|S|=d} |\hat{f}_S| \le \sqrt{\binom{n}{d}\sum_{|S|=d}\hat{f}_S^2} \le n^{d/2}\sqrt{W_{2,d}[f]}. \qquad ◀$$

▶ **Lemma 13.** *Let $f\colon \{0,1\}^n \to [-1,1]$ be a product test of k functions $f_1,\dots,f_k$ with input length m, and $\alpha_i := (1-\mathbb{E}[f_i])/2$ for every $i\in[k]$. Let d be a positive integer. We have*

$$W_{1,d}[f] \le \big(\sqrt{32e^3m}\big)^d g(\alpha_1,\dots,\alpha_k),$$

where the function $g\colon (0,1]^k \to \mathbb{R}$ is defined by

$$g(\alpha_1,\dots,\alpha_k) := e^{-2\sum_{i=1}^k \alpha_i}\sum_{\ell=1}^{d}\sum_{\substack{S\subseteq[k]\\|S|=\ell}}\sum_{\substack{z\in[m]^S\\\sum_i z_i=d}}\prod_{i\in S}\Big(\alpha_i\big(\ln\big(e/\alpha_i^{1/z_i}\big)\big)^{z_i/2}\Big).$$

Proof. For notational simplicity, we will use $W_d[f]$ to denote $W_{1,d}[f]$. Write $f=\prod_{i=1}^k f_i$. Without loss of generality we will assume each function f_i is non-constant. Since f_i and $-f_i$ have the same weight $W_d[f_i]$, we will further assume $\mathbb{E}[f_i]\in[0,1)$. Note that for a subset $S=S_1\times\cdots\times S_k\subseteq(\{0,1\}^m)^k$, we have $\hat{f}_S=\prod_{i=1}^k \hat{f}_{i_{S_i}}$. So,

$$W_d[f] = \sum_{\substack{z\in\{0,\dots,m\}^k\\\sum_i z_i=d}}\prod_{i=1}^k W_{z_i}[f_i] = \sum_{\ell=1}^{d}\sum_{\substack{S\subseteq[k]\\|S|=\ell}}\sum_{\substack{z\in[m]^S\\\sum_i z_i=d}}\Big(\prod_{i\in S}W_{z_i}[f_i]\cdot\prod_{i\notin S}W_0[f_i]\Big).$$

Since $x=1-(1-x)\le e^{-(1-x)}$ for every $x\in\mathbb{R}$, for every subset $S\subseteq[k]$ of size at most d, we have

$$\prod_{i\notin S}W_0[f_i] \le e^{-\sum_{i\notin S}(1-W_0[f_i])} \le e^{-\sum_{i\notin S}(1-W_0[f_i])}\cdot e^{\sum_{i\in S}W_0[f_i]} \le e^d\cdot e^{-\sum_{i=1}^k(1-W_0[f_i])}.$$

Hence,

$$W_d[f] = \sum_{\ell=1}^{d} \sum_{\substack{S\subseteq[k]\\|S|=\ell}} \sum_{\substack{z\in[m]^S\\\sum_i z_i=d}} \Big(\prod_{i\in S} W_{z_i}[f_i] \cdot \prod_{i\notin S} W_0[f_i]\Big)$$

$$\leq e^d \cdot e^{-\sum_{i=1}^k (1-W_0[f_i])} \sum_{\ell=1}^{d} \sum_{\substack{S\subseteq[k]\\|S|=\ell}} \sum_{\substack{z\in[m]^S\\\sum_i z_i=d}} \prod_{i\in S} W_{z_i}[f_i]. \tag{2}$$

Define $f_i' := (1-f_i)/2 \in [0,1]$. Let $\alpha_i := \mathbb{E}[f_i'] = (1-\mathbb{E}[f_i])/2 \in (0,1/2]$. Applying Lemma 10 and Fact 12 to the functions f_i', we have for every subset $S \subseteq [k]$ of size at most d,

$$\sum_{\substack{z\in[m]^S\\\sum_i z_i=d}} \prod_{i\in S} W_{z_i}[f_i'] \leq \sum_{\substack{z\in[m]^S\\\sum_i z_i=d}} \prod_{i\in S} \Big(2m^{z_i/2}\alpha_i\big(2e\ln\big(e/\alpha_i^{1/z_i}\big)\big)^{z_i/2}\Big)$$

$$\leq (\sqrt{8em})^d \sum_{\substack{z\in[m]^S\\\sum_i z_i=d}} \prod_{i\in S} \Big(\alpha_i\big(\ln\big(e/\alpha_i^{1/z_i}\big)\big)^{z_i/2}\Big).$$

Note that for every integer $d \geq 1$, we have $W_d[f_i] = 2W_d[f_i']$. Plugging the bound above into Equation (2), we have

$$W_d[f] \leq (2e)^d \cdot e^{-2\sum_{i=1}^k \alpha_i} \sum_{\ell=1}^{d} \sum_{\substack{S\subseteq[k]\\|S|=\ell}} \sum_{\substack{z\in[m]^S\\\sum_i z_i=d}} \prod_{i\in S} W_{z_i}[f_i'] \leq \big(\sqrt{32e^3m}\big)^d g(\alpha_1,\ldots,\alpha_k),$$

where the function $g\colon (0,1]^k \to \mathbb{R}$ is defined by

$$g(\alpha_1,\ldots,\alpha_k) := e^{-2\sum_{i=1}^k \alpha_i} \sum_{\ell=1}^{d} \sum_{\substack{S\subseteq[k]\\|S|=\ell}} \sum_{\substack{z\in[m]^S\\\sum_i z_i=d}} \prod_{i\in S} \Big(\alpha_i\big(\ln\big(e/\alpha_i^{1/z_i}\big)\big)^{z_i/2}\Big). \qquad \blacktriangleleft$$

We now prove Theorems 3 and 4. For every $(\alpha_1,\ldots,\alpha_k) \in (0,1]^k$, let $\alpha := \sum_{i=1}^k \alpha_i/k \in (0,1]$. We note that the upper bound in Theorem 3 is sufficient to prove Theorem 6.

Proof of Theorem 3. We will bound above $g(\alpha_1,\ldots,\alpha_k)$ in Lemma 13. Recall that $\alpha_i = (1-\mathbb{E}[f_i])/2$. Since $|\mathbb{E}[f_i]| \leq 1-2^{-cm}$, we have $\alpha_i \geq 2^{-(cm+1)}$, and so $\ln(1/\alpha_i) \leq cm+1$. For every subset $S \subseteq [k]$, the set $\{z \in [m]^S : \sum_i z_i = d\}$ has size at most $\binom{d-1}{|S|-1} \leq 2^d$. Hence,

$$\sum_{\substack{z\in[m]^S\\\sum_i z_i=d}} \prod_{i\in S} \big(\ln(1/\alpha_i)\big)^{z_i/2} \leq 2^d(cm+1)^{d/2}.$$

By Maclaurin's inequality (cf. [48, Chapter 12]), we have

$$\sum_{\substack{S\subseteq[k]\\|S|=\ell}} \prod_{i\in S} \alpha_i \leq (e/\ell)^\ell \Big(\sum_{i=1}^k \alpha_i\Big)^\ell = (e/\ell)^\ell (k\alpha)^\ell.$$

Because the function $x \mapsto e^{-2x}x^\ell$ is maximized when $x = \ell/2$, it follows that

$$\sum_{\ell=1}^{d} e^{-2k\alpha} \sum_{\substack{S\subseteq[k]\\|S|=\ell}} \prod_{i\in S} \alpha_i \le \sum_{\ell=1}^{d} e^{-2k\alpha}(e/\ell)^\ell (k\alpha)^\ell \le \sum_{\ell=1}^{d} e^{-\ell}(e/\ell)^\ell(\ell/2)^\ell = \sum_{\ell=1}^{d} 2^{-\ell} \le 1.$$

Therefore,

$$\begin{aligned} g(\alpha_1,\dots,\alpha_k) &= e^{-2\sum_{i=1}^k \alpha_i} \sum_{\ell=1}^{d} \sum_{\substack{S\subseteq[k]\\|S|=\ell}} \sum_{\substack{z\in[m]^S\\\sum_i z_i = d}} \prod_{i\in S} \Big(\alpha_i \big(\ln(1/\alpha_i^{1/z_i})\big)^{z_i/2}\Big) \\ &\le 2^d (cm+1)^{d/2} \sum_{\ell=1}^{d} e^{-2k\alpha} \sum_{\substack{S\subseteq[k]\\|S|=\ell}} \prod_{i\in S} \alpha_i \\ &\le 2^d (cm+1)^{d/2}. \end{aligned}$$

Plugging this bound into Lemma 13, we have

$$W_{1,d}[f] \le \big(\sqrt{32e^3 m}\big)^d \cdot \big(\sqrt{4(cm+1)}\big)^d \le \big(72(\sqrt{c}\cdot m)\big)^d. \qquad \blacktriangleleft$$

We now prove Theorem 4. Recall that we let $\alpha := \sum_{i=1}^k \alpha_i/k \in (0,1]$ for every $(\alpha_1,\dots,\alpha_k) \in (0,1]^k$. We will show that the maximum of the function g defined in Lemma 13 is attained at the diagonal $(\alpha,\dots,\alpha)$. We state the claim now and defer the proof to the next section.

▷ Claim 14. Let g be the function defined in Lemma 13. For every $(\alpha_1,\dots,\alpha_k) \in (0,1]^k$, we have $g(\alpha_1,\dots,\alpha_k) \le g(\alpha,\dots,\alpha)$.

Proof of Theorem 4. We first apply Claim 14 and obtain

$$g(\alpha_1,\dots,\alpha_k) \le g(\alpha,\dots,\alpha) = e^{-2k\alpha} \sum_{\ell=1}^{d} \sum_{\substack{S\subseteq[k]\\|S|=\ell}} \alpha^\ell \sum_{\substack{z\in[m]^S\\\sum_i z_i = d}} \prod_{i\in S} \big(\ln\big(e/\alpha^{1/z_i}\big)\big)^{z_i/2}.$$

We next give an upper bound on $g(\alpha,\dots,\alpha)$ that has no dependence on the numbers z_i. By the weighted AM-GM inequality, for every subset $S \subseteq [k]$ of size ℓ and numbers z_i such that $\sum_{i\in S} z_i = d$,

$$\begin{aligned} \prod_{i\in S} \big(\ln\big(e/\alpha^{1/z_i}\big)\big)^{z_i/2} &\le \Big(\sum_{i\in S} \frac{z_i \ln\big(e/\alpha^{1/z_i}\big)}{d}\Big)^{d/2} \\ &= \Big(\frac{1}{d}\sum_{i\in S} z_i\Big(1 + \frac{1}{z_i}\ln(1/\alpha)\Big)\Big)^{d/2} \\ &= \Big(1 + \frac{\ell}{d}\ln(1/\alpha)\Big)^{d/2} \\ &= \big(\ln\big(e/\alpha^{\ell/d}\big)\big)^{d/2}. \end{aligned}$$

For every subset $S \subseteq [k]$, the set $\{z \in [m]^S : \sum_i z_i = d\}$ has size at most $\binom{d-1}{|S|-1} \leq 2^d$. Thus,

$$\begin{aligned} g(\alpha,\dots,\alpha) &\leq e^{-2k\alpha} \sum_{\ell=1}^{d} \sum_{\substack{S\subseteq[k]\\|S|=\ell}} \alpha^\ell \sum_{\substack{z\in[m]^S\\ \sum_i z_i = d}} \big(\ln\big(e/\alpha^{\ell/d}\big)\big)^{d/2} \\ &\leq 2^d \sum_{\ell=1}^{d} e^{-2k\alpha} \sum_{\substack{S\subseteq[k]\\|S|=\ell}} \alpha^\ell \big(\ln\big(e/\alpha^{\ell/d}\big)\big)^{d/2} \\ &\leq 2^d \sum_{\ell=1}^{d} e^{-2k\alpha} \Big(\frac{ek\alpha}{\ell}\Big)^\ell \big(\ln\big(e/\alpha^{\ell/d}\big)\big)^{d/2}. \end{aligned} \tag{3}$$

For every $\ell \in [k]$, define $g_\ell \colon (0,1] \to \mathbb{R}$ to be

$$g_\ell(x) := e^{-2kx} \Big(\frac{ekx}{\ell}\Big)^\ell \big(\ln\big(e/x^{\ell/d}\big)\big)^{d/2}.$$

We now bound above the maximum of g_ℓ over $x \in (0,1]$. One can verify easily that the derivative of g is

$$g'_\ell(x) = \frac{g_\ell(x)}{2x \ln\big(e/x^{\ell/d}\big)} \big(\ln(1/x^{2\ell/d})(\ell - 2kx) + (\ell - 4kx)\big).$$

Observe that when $x \leq \ell/4k$, then $g'_\ell(x) \geq \frac{g_\ell(x)}{4x \ln(e/x^{\ell/d})}\big(\ell \ln(1/x^{2\ell/d})\big) \geq 0$. Likewise, when $x \geq \ell/2k$, then $g'_\ell(x) \leq \frac{g_\ell(x)}{2x \ln(e/x^{\ell/d})}(-\ell) \leq 0$. Also, we have $g_\ell(0) = 0$. Hence, $g_\ell(x) \leq g_\ell(\beta_\ell \ell/4k)$ for some $\beta_\ell \in [1,2]$, which is at most

$$e^{-\ell/2} \cdot (e/2)^\ell \cdot \Big(\ln\big(e(4k/\ell)^{\ell/d}\big)\Big)^{d/2}.$$

(In the case when $\ell/4k \geq 1$, we have $g_\ell(x) \leq g_\ell(1) \leq e^{-2k}(ek/\ell)^\ell$.) Therefore, plugging this back into Equation (3),

$$\begin{aligned} g(\alpha,\dots,\alpha) \leq 2^d \sum_{\ell=1}^{d} g_\ell(\alpha) &\leq 2^d \sum_{\ell=1}^{d} g_\ell(\beta_\ell \ell/4k) \\ &\leq 2^d \sum_{\ell=1}^{d} e^{-\ell/2} \cdot (e/2)^\ell \cdot \Big(\ln\big(e(4k/\ell)^{\ell/d}\big)\Big)^{d/2} \\ &\leq 2^d \big(e \ln(4ek)\big)^{d/2} \sum_{\ell=1}^{d} 2^{-\ell} \\ &\leq \big(\sqrt{4e \ln(4ek)}\big)^d. \end{aligned}$$

Putting this back into the bound in Lemma 13, we conclude that

$$W_{1,d}[f] \leq \big(84\sqrt{m \ln(4ek)}\big)^d,$$

proving the theorem. ◀

2.1 Schur-concavity of g

We prove Claim 14 in this section. First recall that the function $g\colon (0,1]^k \to \mathbb{R}$ is defined as

$$g(\alpha_1,\dots,\alpha_k) := \sum_{\ell=1}^{d} \sum_{\substack{S\subseteq[k]\\|S|=\ell}} \sum_{\substack{z\in[m]^S\\ \sum_i z_i = d}} \prod_{i\in S} \phi_{z_i}(\alpha_i),$$

where for every positive integer z, the function $\phi_z\colon (0,1] \to \mathbb{R}$ is defined by

$$\phi_z(x) = x\ln(e/x^{1/z})^{z/2}.$$

The proof of Claim 14 follows from showing that g is *Schur-concave*. Before defining it, we first recall the concept of majorization. Let $x, y \in \mathbb{R}^k$ be two vectors. We say that y *majorizes* x, denoted by $x \prec y$, if for every $j \in [k]$ we have

$$\sum_{i=1}^{j} x_{(i)} \le \sum_{i=1}^{j} y_{(i)},$$

and $\sum_{i=1}^{k}(x_i - y_i) = 0$, where $x_{(i)}$ and $y_{(i)}$ are the ith largest coordinates in x and y respectively.

A function $f\colon D \to \mathbb{R}$ where $D \subseteq \mathbb{R}^k$ is *Schur-concave* if whenever $x \prec y$ we have $f(x) \ge f(y)$. We will show that g is Schur-concave using the Schur–Ostrowski criterion.

▶ **Theorem 15** (Schur–Ostrowski criterion (Theorem 12.25 in [43])). *Let $f\colon D \to \mathbb{R}$ be a function where $D \subseteq \mathbb{R}^k$ is permutation-invariant, and assume that the first partial derivatives of f exist in D. Then f is Schur-concave in D if and only if*

$$(x_j - x_i)\Big(\frac{\partial f}{\partial x_i} - \frac{\partial f}{\partial x_j}\Big) \ge 0$$

for every $x \in D$, and every $1 \le i \ne j \le k$.

Claim 14 then follows from the observation that $(\sum_i x_i/k, \dots, \sum_i x_i/k) \prec x$ for every $x \in [0,1]^k$.

▷ Claim 16. For every $x \in (0,1]$ we have

1. $\phi_z(x) \ge 0$;
2. $\phi_z'(x) = \frac{1}{2}\ln\big(\frac{e}{x^{2/z}}\big)\ln\big(\frac{e}{x^{1/z}}\big)^{z/2-1} > 0$, and
3. $\phi_z''(x) = -\frac{1}{2xz}\ln\big(\frac{e}{x^{1/z}}\big)^{z/2-2}\big(2\ln\big(\frac{e}{x^{1/z}}\big) + (\frac{z}{2}-1)\ln\big(\frac{e}{x^{2/z}}\big)\big) \le 0.$

Proof. The derivatives of ϕ_z and the non-negativity of ϕ_z and ϕ_z' can be verified easily. It is also clear that ϕ_z'' is non-positive when $z \ge 2$. Thus it remains to verify $\phi_1''(x) \le 0$ for every x. We have

$$\phi_1''(x) = -\frac{1}{2x}\ln\Big(\frac{e}{x}\Big)^{-3/2}\Big(2\ln\Big(\frac{e}{x}\Big) - \frac{1}{2}\ln\Big(\frac{e}{x^2}\Big)\Big).$$

It follows from $\frac{1}{2}\ln(e/x^2) \le \ln(e^2/x^2) = 2\ln(e/x)$ that $\phi_1''(x) \le 0$. ◁

▶ **Lemma 17.** *g is Schur-concave.*

Proof. Fix $1 \le u \ne v \le k$ and write $g = g_1 + g_2$, where

$$g_1(\alpha_1, \ldots, \alpha_k) := \sum_{\ell=1}^{d} \sum_{\substack{S \subseteq [k], |S|=\ell \\ (S \ni u \wedge S \not\ni v) \vee (S \not\ni u \wedge S \ni v)}} \sum_{\substack{z \in [m]^S \\ \sum_i z_i = d}} \prod_{i \in S} \phi_{z_i}(\alpha_i)$$

and

$$g_2(\alpha_1, \ldots, \alpha_k) := \sum_{\ell=1}^{d} \sum_{\substack{S \subseteq [k], |S|=\ell \\ (S \ni u \wedge S \ni v) \vee (S \not\ni u \wedge S \not\ni v)}} \sum_{\substack{z \in [m]^S \\ \sum_i z_i = d}} \prod_{i \in S} \phi_{z_i}(\alpha_i).$$

We will show that for every $\alpha \in (0,1]^k$, whenever $\alpha_v \le \alpha_u$ we have (1) $\left(\frac{\partial g_1}{\partial \alpha_u} - \frac{\partial g_1}{\partial \alpha_v}\right)(\alpha) \le 0$ and (2) $\left(\frac{\partial g_2}{\partial \alpha_u} - \frac{\partial g_2}{\partial \alpha_v}\right)(\alpha) \le 0$, from which the lemma follows from Theorem 15.

For g_1, since $\phi_z'' \le 0$ and $\alpha_v \le \alpha_u$, we have $\phi'_{z_u}(\alpha_v) \ge \phi'_{z_u}(\alpha_u)$. Moreover, as $\phi_z \ge 0$ and $\phi'_z > 0$, we have

$$\begin{aligned}
\frac{\partial g_1}{\partial \alpha_u}(\alpha) &\le \sum_{\ell=1}^{d} \sum_{\substack{S \subseteq [k], |S|=\ell \\ (S \ni u \wedge S \not\ni v)}} \sum_{\substack{z \in [m]^S \\ \sum_i z_i = d}} \prod_{\substack{i \in S \\ i \ne u}} \phi_{z_i}(\alpha_i) \cdot \phi'_{z_u}(\alpha_u) \cdot \frac{\phi'_{z_u}(\alpha_v)}{\phi'_{z_u}(\alpha_u)} \\
&= \sum_{\ell=1}^{d} \sum_{\substack{S \subseteq [k], |S|=\ell \\ (S \ni u \wedge S \not\ni v)}} \sum_{\substack{z \in [m]^S \\ \sum_i z_i = d}} \prod_{\substack{i \in S \\ i \ne u}} \phi_{z_i}(\alpha_i) \cdot \phi'_{z_u}(\alpha_v) \\
&= \sum_{\ell=1}^{d} \sum_{\substack{S \subseteq [k], |S|=\ell \\ (S \ni v \wedge S \not\ni u)}} \sum_{\substack{z \in [m]^S \\ \sum_i z_i = d}} \prod_{\substack{i \in S \\ i \ne v}} \phi_{z_i}(\alpha_i) \cdot \phi'_{z_v}(\alpha_v) = \frac{\partial g_1}{\partial \alpha_v}(\alpha),
\end{aligned}$$

where in the second equality we simply renamed z_u to z_v.

We now show that $\left(\frac{\partial g_2}{\partial \alpha_u} - \frac{\partial g_2}{\partial \alpha_v}\right)(\alpha) \le 0$ whenever $\alpha_v \le \alpha_u$. For all positive integers z and w, define $\psi_{z,w} \colon (0,1]^2 \to \mathbb{R}$ by

$$\psi_{z,w}(x,y) := \phi'_z(x)\phi_w(y) + \phi'_w(x)\phi_z(y) - \phi_z(x)\phi'_w(y) - \phi_w(x)\phi'_z(y).$$

Note that when $x = y$ we have $\psi_{z,w}(x,x) = 0$. Moreover, when $z = w$ we have $\psi_{z,z}(x,y) = 2(\phi'_z(x)\phi_z(y) - \phi_z(x)\phi'_z(y))$. For every $x, y \in (0,1]$, by Claim 16 we have

$$\frac{\partial}{\partial y}\psi_{z,w}(x,y) = \phi'_z(x)\phi'_w(y) + \phi'_w(x)\phi'_z(y) - \phi_z(x)\phi''_w(y) - \phi_w(x)\phi''_z(y) \ge 0.$$

Since $\psi_{z_u,z_v}(\alpha_u, \alpha_u) = 0$, we have $\psi_{z_u,z_v}(\alpha_u, \alpha_v) \le 0$ whenever $\alpha_v \le \alpha_u$, and so

$$\begin{aligned}
&\left(\frac{\partial g_2}{\partial \alpha_u} - \frac{\partial g_2}{\partial \alpha_v}\right)(\alpha) = \\
&\sum_{\ell=2}^{d} \sum_{\substack{S \subseteq [k] \\ |S|=\ell \\ S \ni u \wedge S \ni v}} \Bigg(\sum_{\substack{z \in [m]^S \\ \sum_i z_i = d \\ z_u = z_v}} \prod_{\substack{i \in S \\ i \ne u \\ i \ne v}} \phi_{z_i}(\alpha_i) \cdot \psi_{z_u,z_v}(\alpha_u, \alpha_v)/2 + \sum_{\substack{z \in [m]^S \\ \sum_i z_i = d \\ z_u < z_v}} \prod_{\substack{i \in S \\ i \ne u \\ i \ne v}} \phi_{z_i}(\alpha_i) \cdot \psi_{z_u,z_v}(\alpha_u, \alpha_v) \Bigg) \le 0
\end{aligned}$$

because the values ϕ_{z_i} are non-negative. ◀

2.2 Lower bound

In this section we prove Claim 5. We first restate our claim.

▷ Claim 5. For all positive integers m and d, there exists a product test $f\colon \{0,1\}^{mk} \to \{0,1\}$ with $k = d \cdot 2^m$ functions of input length m such that

$$W_{1,d}[f] \geq (m/e^{3/2})^d.$$

Proof. Let $k = d \cdot 2^m$ and $f_1, \ldots, f_k\colon \{0,1\}^{mk} \to \{0,1\}$ be the OR function on k disjoint sets of m bits. It is easy to verify that $\hat{f}_i(\emptyset) = 1 - 2^{-m}$ and $|\hat{f}_i(S)| = 2^{-m}$ for every $S \neq \emptyset$. Consider the product test $f := \prod_{i=1}^{k} f_i$. Using the fact that $1 - x \geq e^{-x(1+x)}$ for $x \in [0, 1/2]$, we have

$$(1 - 2^{-m})^k \geq e^{-2^m(1+2^{-m})k} \geq e^{-d(1+2^{-m})} \geq e^{-3d/2}.$$

Hence,

$$\begin{aligned}
W_{1,d}[f] &= \sum_{\substack{z \in \{0,\ldots,m\}^k \\ \sum_i z_i = d}} \prod_{i=1}^{k} W_{z_i}[f_i] \\
&\geq \sum_{|S|=d} \Big(\prod_{i \in S} W_{1,1}[f_i] \prod_{i \notin S} W_{1,0}[f_i] \Big) \\
&= \binom{k}{d} \cdot (m2^{-m})^d \cdot (1 - 2^{-m})^{k-d} \\
&\geq \Big(\frac{d \cdot 2^m}{d} \Big)^d \cdot (m2^{-m})^d \cdot e^{-3d/2} \\
&= (m/e^{3/2})^d.
\end{aligned}$$

◀

3 Pseudorandom generators

In this section, we use Theorem 4 to construct two pseudorandom generators for product tests. The first one (Theorem 8) has seed length $\tilde{O}(m + \log(k/\varepsilon)) \log k$. The second one (Theorem 6) has a seed length of $\tilde{O}(m + \log(n/\varepsilon))$ but only works for product tests with outputs $\{-1, 1\}$ and their variants (see Corollary 7). We note that Theorem 6 can also be obtained using Theorem 3 in place of Theorem 4.

Both constructions use the Ajtai–Wigderson framework [5, 23], and follow from recursively applying the following theorem, which roughly says that $2^{-\tilde{\Omega}(m+\log(k/\varepsilon))}$-almost $O(m + \log(k/\varepsilon))$-wise independence plus constant fraction of noise fools product tests.

▶ **Theorem 11.** *Let $f\colon \{0,1\}^n \to [-1,1]$ be a product test with k functions of input length m. Let d be a positive integer. Let D and T be two independent δ-almost d-wise independent distributions over $\{0,1\}^n$, and U be the uniform distribution over $\{0,1\}^n$. Then*

$$\big|\mathbb{E}[f(D + T \wedge U)] - \mathbb{E}[f(U)]\big| \leq k \cdot \big(\sqrt{\delta} \cdot (170 \cdot \sqrt{m \ln(ek)})^d + 2^{-(d-m)/2}\big),$$

where "$+$" and "$\wedge$" are bit-wise XOR and AND respectively.

Theorem 11 follows immediately by combining Theorem 4 and Lemma 18 below.

▶ **Lemma 18.** *Let $f\colon \{0,1\}^n \to [-1,1]$ be a product test with k functions of input length m. Let d be a positive integer. Let D, T, U be a δ-almost $(d+m)$-wise independent, a γ-almost $(d+m)$-wise independent, and the uniform distributions over $\{0,1\}^n$, respectively. Then*

$$\left|\mathbb{E}[f(D+T\wedge U)] - \mathbb{E}[f(U)]\right| \le k\cdot\left(\sqrt{\delta}\cdot W_{1,\le d+m}[f] + 2^{-d/2} + \sqrt{\gamma}\right),$$

where "$+$" and "$\wedge$" are bit-wise XOR and AND respectively.

Proof. We slightly modify the decomposition in [21, Proposition 6.1] as follows. Let f be a product test and write $f = \prod_{i=1}^{k} f_i$. As the distribution $D + T \wedge U$ is symmetric, we can assume the function f_i is defined on the ith m bits. For every $i \in \{1,\dots,k\}$, let $f^{\le i} = \prod_{j\le i} f_j$ and $f^{>i} = \prod_{j>i} f_j$. We decompose f into

$$f = \hat{f}_\emptyset + L + \sum_{i=1}^{k} H_i f^{>i}, \tag{4}$$

where

$$L := \sum_{\substack{\alpha\in\{0,1\}^{mk}\\ 0<|\alpha|<d}} \hat{f}_\alpha \chi_\alpha$$

and

$$H_i := \sum_{\substack{\alpha=(\alpha_1,\dots,\alpha_i)\in\{0,1\}^{mi}:\\ \text{the } d\text{th 1 in } \alpha \text{ appears in } \alpha_i}} \hat{f}^{\le i}_\alpha \chi_\alpha.$$

We now show that the expressions on both sides of Equation (4) are identical. Clearly, every Fourier coefficient on the right hand side is a coefficient of f. To see that every coefficient of f appears on the right hand side exactly once, let $\alpha = (\alpha_1,\dots,\alpha_k) \in \{0,1\}^{mk}$ and $\hat{f}_\alpha = \prod_{i=1}^{k} \hat{f}_i(\alpha_i)$ be a coefficient of f. If $|\alpha| < d$, then $\hat{f}_\alpha$ appears in $\hat{f}_\emptyset$ or L. Otherwise, $|\alpha| \ge d$. Then the dth 1 in α must appear in one of $\alpha_1,\dots,\alpha_k$. Say it appears in α_i. Then we claim that α appears in $H_i f^{>i}$. This is because the coefficient indexed by $(\alpha_1,\dots,\alpha_i)$ appears in H_i, and the coefficient indexed by $(\alpha_{i+1},\dots,\alpha_k)$ appears in $f^{>i}$. Note that all the coefficients in each function H_i have weights between d and $d+m$, and because our distributions D and T are both almost $(d+m)$-wise independent, we get an error of $2^{-d}+\gamma$ in Lemma 7.1 in [21]. The rest of the analysis follows from [21] or [26]. ◀

3.1 Generator for product tests

We now prove Theorem 8.

▶ **Theorem 8.** *There exists an explicit generator $G\colon \{0,1\}^\ell \to \{0,1\}^n$ that fools any product test with k functions of input length m with error ε and seed length $O(\log mk)((m + \log(k/\varepsilon))(\log m + \log\log(k/\varepsilon)) + \log\log n) = \tilde{O}(m + \log(k/\varepsilon))\log k$.*

The high-level idea is very simple. Let f be a product test. For every choice of D and T in Theorem 11, the function $f'\colon \{0,1\}^T \to [-1,1]$ defined by $f'(y) := f(D + T \wedge y)$ is also a product test. So we can apply Theorem 11 again and recurse. We show that if we repeat this argument for $t = O(\log(mk))$ times with t independent copies of D and T, then for every fixing of $D_1,\dots,D_t$ and with high probability over the choice of $T_1,\dots,T_t$, the restricted product test defined on $\{0,1\}^{\bigwedge_{i=1}^{t} T_i}$ is a product test defined on at most $O(m + \log(k/\varepsilon))$ bits, which can then be fooled by an almost $O(m + \log(k/\varepsilon))$-wise independent distribution.

Proof of Theorem 8. Let C be a sufficiently large constant. Let $d = C(m + \log(k/\varepsilon))$, $\delta = d^{-2d}$, and $t = C\log(mk) = \tilde{O}(\log k)$. Let $D_1, \dots, D_t, T_1, \dots, T_t$ be $2t$ independent δ-almost d-wise independent distributions over $\{0,1\}^n$. Define $D^{(1)} := D_1$ and $D^{(i+1)} := D_{i+1} + T_i \wedge D^{(i)}$.

Let $D := D^{(t)}$, $T := \bigwedge_{i=1}^{t} T_i$. Let G' be a δ-almost d-wise independent distribution over $\{0,1\}^n$. For a subset $S \subseteq [n]$, define the function $\mathrm{PAD}_S(x)\colon \{0,1\}^{|S|} \to \{0,1\}^n$ to output n bits of which the positions in S are the first $|S|$ bits of $x0^{|S|}$ and the rest are 0. Our generator G outputs

$$D + T \wedge \mathrm{PAD}_T(G').$$

We first look at the seed length of G. By [39, Lemma 4.2], sampling the distributions D_i and T_i takes a seed of length

$$\begin{aligned} s &:= t \cdot O(d\log d + \log\log n) \\ &= t \cdot O\big((m + \log(k/\varepsilon))(\log m + \log\log(k/\varepsilon)) + \log\log n\big) \\ &= t \cdot \tilde{O}\big(m + \log(k/\varepsilon)\big). \end{aligned}$$

Sampling G' takes a seed of length $O((m+\log(k/\varepsilon))(\log m+\log\log(k/\varepsilon))+\log\log n)$. Hence the total seed length of G is $\tilde{O}(m + \log(k/\varepsilon))\log k$.

We now look at the error of G. By our choice of δ and applying Theorem 11 recursively for t times, we have

$$\begin{aligned} \big|\mathbb{E}[f(D + T \wedge U)] - \mathbb{E}[f(U)]\big| &\le t \cdot k \cdot \Big(\sqrt{\delta} \cdot \big(170 \cdot \sqrt{m\ln(ek)}\big)^d + 2^{-(d-m)/2}\Big) \\ &\le t \cdot k \cdot \Big(\Big(\frac{170\sqrt{m\ln(ek)}}{d}\Big)^d + 2^{-\Omega(d)}\Big) \\ &\le t \cdot 2^{-\Omega(d)} \le \varepsilon/2. \end{aligned}$$

Next, we show that for every fixing of D and most choices of T, the function $f_{D,T}(y) := f(D + T \wedge y)$ is a product test defined on d bits, which can be fooled by G'.

Let $I = \bigcup_{i=1}^{k} I_i$. Note that $|I| \le mk$. Because the variables T_i are independent and each of them is δ-almost d-wise independent, we have

$$\Pr\big[|I \cap T| \ge d\big] \le \binom{|I|}{d}(2^{-d} + \delta)^t \le 2^{d\log(mk)} \cdot 2^{-\Omega(d\log(mk))} \le \varepsilon/4.$$

It follows that for every fixing of D, with probability at least $1 - \varepsilon/4$ over the choice of T, the function $f_{D,T}$ is a product test defined on at most d bits, which can be fooled by G' with error $\varepsilon/4$. Hence G fools f with error ε. ◀

3.2 Almost-optimal generator for XOR of Boolean functions

In this section, we construct our generator for product tests with outputs $\{-1, 1\}$, which correspond to the XOR of Boolean functions f_i defined on disjoint inputs. Throughout this section we will call these tests $\{-1,1\}$-*products*. We first restate our theorem.

▶ **Theorem 6.** *There exists an explicit generator $G\colon \{0,1\}^\ell \to \{0,1\}^n$ that fools the XOR of any k Boolean functions on disjoint inputs of length $\le m$ with error ε and seed length $O(m + \log(n/\varepsilon))(\log m + \log\log(n/\varepsilon))^2 = \tilde{O}(m + \log(n/\varepsilon))$.*

Theorem 6 relies on applying the following lemma recursively in different ways. From now on, we will relax our tests to allow one of the k functions to have input length greater than m, but bounded by $O(m + \log(n/\varepsilon))$.

▶ **Lemma 19.** *There exists a constant C such that the following holds. Let m and s be two integers such that $m \geq C \log\log(n/\varepsilon)$ and $s = 5(m+\log(n/\varepsilon))$. If there is an explicit generator $G' \colon \{0,1\}^{\ell'} \to \{0,1\}^n$ that fools $\{-1,1\}$-products with $k' \leq 16^{m+1}$ functions, $k'-1$ of which have input lengths $\leq m/2$ and one has length $\leq s$, with error ε' and seed length ℓ', then there is an explicit generator $G \colon \{0,1\}^{\ell} \to \{0,1\}^n$ that fools $\{-1,1\}$-products with $k \leq 16^{2m+1}$ functions, $k-1$ of which have input lengths $\leq m$ and one has length $\leq s$, with error $\varepsilon' + \varepsilon$ and seed length $\ell = \ell' + O(m + \log(n/\varepsilon))(\log m + \log\log(n/\varepsilon)) = \ell' + \tilde{O}(m + \log(n/\varepsilon))$.*

The proof of Lemma 19 closely follows a construction by Meka, Reingold and Tal [38]. First of all, we will use the following generator in [38]. It fools any $\{-1,1\}$-products when the number of functions k is significantly greater than the input length m of the functions f_i.

▶ **Lemma 20** (Lemma 6.2 in [38])**.** *There exists a constant C such that the following holds. Let n, k, m, s be integers such that $C \log\log(n/\varepsilon) \leq m \leq \log n$ and $16^m \leq k \leq 2 \cdot 16^{2m}$. There exists an explicit pseudorandom generator $G_{\oplus\mathrm{Many}} \colon \{0,1\}^{\ell} \to \{0,1\}^n$ that fools $\{-1,1\}$-products with k non-constant functions, $k-1$ of which have input lengths $\leq m$ and one has length $\leq s$, with error ε and seed length $O(s + \log(n/\varepsilon))$.*

Here is the high-level idea of proving Lemma 19. We consider two cases depending on whether k is large with respect to m. If $k \geq 16^m$, then by Lemma 20, the generator $G_{\oplus\mathrm{Many}}$ fools f. Otherwise, we show that for every fixing of D and most choices of T, the restriction of f under (D,T) is a $\{-1,1\}$-product with k functions, $k-1$ of which have input length $\leq m/2$ and one has length $\leq s$. More specifically, we will show that for most choices of T, the following would happen: for the function with input length $\leq s$, at most $s/2$ of its inputs remain in T; for the rest of the functions with input length $\leq m$, after being restricted by (D,T), at most $\lceil s/2m \rceil$ of them have input length $> m/2$, and so they are defined on a total of $s/2$ positions in T. Now we can think of these "bad" functions as one function with input length $\leq s$, and the rest of the at most k "good" functions have input length $m/2$. So we can apply the generator G' in our assumption.

Proof of Lemma 19. Let C be the constant in Lemma 20 and C' be a sufficiently large constant.

Let $d = C's$ and $\delta = d^{-2d}$. Let $D_1, \ldots, D_{50}, T_1, \ldots, T_{50}$ be 100 independent δ-almost d-wise independent distributions over $\{0,1\}^n$. Define $D^{(1)} := D_1$ and $D^{(i+1)} := D_{i+1} + T_i \wedge D^{(i)}$.

Let $D := D^{(50)}$, $T := \bigwedge_{i=1}^{50} T_i$ and $G_{\oplus\mathrm{Many}}$ be the generator in Lemma 20 with respect to the values of n, k, m, s given in this lemma. For a subset $S \subseteq [n]$, define the function $\mathrm{PAD}_S(x) \colon \{0,1\}^{|S|} \to \{0,1\}^n$ to output n bits of which the positions in S are the first $|S|$ bits of $x0^{|S|}$ and the rest are 0. Our generator G outputs

$$(D + T \wedge \mathrm{PAD}_T(G')) + G_{\oplus\mathrm{Many}}.$$

We first look at the seed length of G. By Lemma 20, $G_{\oplus\mathrm{Many}}$ uses a seed of length $O(s + \log(n/\varepsilon)) = O(m + \log(n/\varepsilon))$. By [39, Lemma 4.2], sampling the distributions D_i and T_i takes a seed of length

$$O(s \log s) = O\big(m + \log(n/\varepsilon)\big)(\log m + \log\log(n/\varepsilon)) = \tilde{O}(m + \log(n/\varepsilon)).$$

Hence the total seed length of G is $\ell' + O(m + \log(n/\varepsilon))(\log m + \log\log(n/\varepsilon)) = \ell' + \tilde{O}(m + \log(n/\varepsilon))$.

We now show that G fools f. Write $f = \prod_{i=1}^{k} f_i$, where $f_i \colon \{0,1\}^{I_i} \to \{-1,1\}$. Without loss of generality we can assume each function f_i is non-constant. We consider two cases.

k is large

If $k \geq 16^m$, then for every fixing of D, T and G', the function $f'(y) := f(D+T\wedge \mathrm{PAD}_T(G')+y)$ is also a $\{-1,1\}$-product with the same parameters as f. Note that we always have $k \leq n$ and so $m \leq \log n$. Hence it follows from Lemma 20 that the generator $G_{\oplus\mathrm{Many}}$ fools f' with error ε. Averaging over D, T and G' shows that G fools f with error ε.

k is small

Now suppose $k \leq 16^m$. For every fixing of $G_{\oplus\mathrm{Many}}$, consider $f'(y) := f(y + G_{\oplus\mathrm{Many}})$. Again, f' is a $\{-1,1\}$-product with the same parameters as f. In particular, it is a $\{-1,1\}$-product with k functions with input length s. So, by our choice of δ and applying Theorem 11 recursively for 50 times, we have

$$\begin{aligned}\left|\mathbb{E}[f'(D+T\wedge U)] - \mathbb{E}[f'(U)]\right| &\leq 50\cdot k\cdot \left(\sqrt{\delta}\cdot\left(170\cdot\sqrt{s\ln(ek)}\right)^d + 2^{-(d-s)/2}\right)\\ &\leq 50\cdot 2^s\cdot\left((170s/d)^d + 2^{-\Omega(s)}\right)\\ &\leq 2^{-\Omega(s)} \leq \varepsilon/2.\end{aligned}$$

Next, we show that for every fixing of D and most choices of T, the function $f'_{D,T}(y) := f'(D+T\wedge y)$ is a $\{-1,1\}$-product with k functions, $k-1$ of which have input lengths $\leq m/2$ and one has length $\leq s$, which can be fooled by G'.

Because the variables T_i are independent and each of them is δ-almost d-wise independent, for every subset $I \subseteq [n]$ of size at most d, we have

$$\Pr[T\cap I = I] = \prod_{i=1}^{50}\Pr[T_i\cap I = I] \leq (2^{-|I|}+\delta)^{50} \leq (3/4)^{-50|I|}.$$

Without loss of generality, we assume $I_1,\ldots,I_{k-1}$ are the subsets of size at most m and I_k is the subset of size at most s. We now look at which subsets $T\cap I_i$ have length at most $m/2$ and which subsets do not. For the latter, we collect the indices in these subsets.

Let $G := \{i\in[k-1] : |T\cap I_i| \leq m/2\}$, $B := \{i\in[k-1] : |T\cap I_i| > m/2\}$ and $BV := \{j\in[n] : j\in\bigcup_{i\in B}(T\cap I_i)\}$. We claim that with probability $1-\varepsilon/2$ over the choice of T, we have $|BV|\leq s$. Note that the indices in BV either come from I_k, or I_i for $i\in[k-1]$. For the first case, the probability that at least $s/2$ of the indices in I_k appear in BV is at most

$$\binom{|I_k|}{s/2}(3/4)^{-25s} \leq 2^s\cdot(3/4)^{-25s} \leq \varepsilon/4.$$

For the second case, note that if at least $s/2$ of the variables in $\bigcup_{i\in[k-1]} I_i$ appear in BV, then they must appear in at least $\lceil s/2m\rceil$ of the subsets $T\cap I_1,\ldots,T\cap I_{k-1}$. The probability of the former is at most the probability of the latter, which is at most

$$\binom{k-1}{\lceil s/2m\rceil}\binom{m\cdot\lceil s/2m\rceil}{s/2}(3/4)^{-25s} \leq 16^{m\cdot(s/2m+1)}\cdot 2^{m\cdot(s/2m+1)}\cdot(3/4)^{-25s} \leq \varepsilon/4,$$

because $k\leq 16^m$ and $m\leq s$. Hence with probability $1-\varepsilon/2$ over the choice of T, the function $f'_{D,T}$ is a product $g\cdot h$, where g is a product of $|G|\leq k-1$ functions of input length

$m/2$, and h is a product of $|B|+1$ functions defined on a total of $|BV| \le s$ bits. Recall that $k \le 16^m$, so by our assumption G' fools $f'_{D,T}$ with error ε'. Therefore G fools f with error $\varepsilon + \varepsilon'$. ◀

We obtain Theorem 6 by applying Lemma 19 repeatedly in different ways.

Proof of Theorem 6. Given a $\{-1,1\}$-product $f\colon \{0,1\}^n \to \{-1,1\}$ with k functions of input length m, we will apply Lemma 19 in stages. In each stage, we start with a $\{-1,1\}$-product f with k_1 functions, $k_1 - 1$ of which have input lengths $\le m_1 = \max\{m, 2\log(n/\varepsilon)\}$ and one has length $\le s := 5(m + \log(n/\varepsilon))$. Note that $k_1 \le 16^{2m_1+1}$. Let C be the constant in Lemma 19. We apply Lemma 19 for $t = O(\log m_1)$ times until f is restricted to a $\{-1,1\}$-product f' with k_2 functions, $k_2 - 1$ of which have input lengths $\le m_2$ and one has length $\le s$, where $m_2 = C\log\log(n/\varepsilon)$, $k_2 \le 16^{2m_2+1} \le (\log(n/\varepsilon))^r$, and $r := 8C + 4$ is a constant. This uses a seed of length

$$\begin{aligned} t \cdot O(m + \log(n/\varepsilon))(\log m + \log\log(n/\varepsilon)) &\le O(m + \log(n/\varepsilon))(\log m + \log\log(n/\varepsilon))^2 \\ &= \tilde{O}(m + \log(n/\varepsilon)). \end{aligned}$$

At the end of each stage, we repeat the above argument by grouping every $\lceil \log(n/\varepsilon)/m_2 \rceil$ functions of f' that have input lengths $\le m_2$ as one function of input length $\le 2\log(n/\varepsilon)$, so we can think of f' as a $\{-1,1\}$-product with $k_3 := k_2/\lceil m_2/(\log n)\rceil \le (\log(n/\varepsilon))^{r-1}\log\log n$ functions, $k_3 - 1$ of which have input lengths $\le \log(n/\varepsilon)$ and one has length $\le s$.

Repeating above for $r + 1 = O(1)$ stages, we are left with a $\{-1,1\}$-product of two functions, one has input length $\le C\log\log(n/\varepsilon)$, and one has length $\le s$, which can then be fooled by a $2^{-\Omega(s)}$-biased distribution that can be sampled using $O(m + \log(n/\varepsilon))$ bits [39]. So the total seed length is $O(m + \log(n/\varepsilon))(\log m + \log\log(n/\varepsilon))^2 = \tilde{O}(m + \log(n/\varepsilon))$, and the error is $(r+1)\cdot t \cdot \varepsilon$. Replacing ε with $\varepsilon/(r+1)t$ proves the theorem. ◀

4 Level-d inequalities

In this section, we prove Lemma 10 that gives an upper bound on the dth level Fourier weight of a $[0,1]$-valued function in L_2-norm. We first restate the lemma.

▶ **Lemma 10.** *Let $g\colon \{0,1\}^n \to [0,1]$ be any function. For every positive integer d, we have*

$$W_{2,d}[g] \le 4\,\mathbb{E}[g]^2\big(2e\ln(e/\mathbb{E}[g]^{1/d})\big)^d.$$

Our proof closely follows the argument in [52].

▷ Claim 21. Let $f\colon \{0,1\}^n \to \mathbb{R}$ have Fourier degree at most d and $\|f\|_2 = 1$. Let $g\colon \{0,1\}^n \to [0,1]$ be any function. If $t_0 \ge 2e^{d/2}$, then

$$\mathbb{E}\big[g(x)|f(x)|\big] \le \mathbb{E}[g]t_0 + 2et_0^{1-2/d}e^{-\frac{d}{2e}t_0^{2/d}}.$$

To prove this claim, we will use the following concentration inequality for functions with Fourier degree d from [18].

▶ **Theorem 22** (Lemma 2.2 in [18])**.** *Let $f\colon \{0,1\}^n \to \mathbb{R}$ have Fourier degree at most d and assume that $\|f\|_2 := \sum_S \hat{f}_S^2 = 1$. Then for any $t \ge (2e)^{d/2}$,*

$$\Pr\big[|f| \ge t\big] \le e^{-\frac{d}{2e}t^{2/d}}.$$

We also need to bound above the integral of $e^{-\frac{d}{2e}t^{2/d}}$.

▷ Claim 23. Let d be any positive integer. If $t_0 \geq (2e)^{d/2}$, then we have

$$\int_{t_0}^{\infty} e^{-\frac{d}{2e}t^{2/d}}dt \leq 2et_0^{1-2/d}e^{-\frac{d}{2e}t_0^{2/d}}.$$

Proof. First we apply the following change of variable to the integral. We set $s = \frac{d}{2e}t^{2/d}$ and obtain

$$\int_{t_0}^{\infty} e^{-\frac{d}{2e}t^{2/d}}dt = e\Big(\frac{2e}{d}\Big)^{d/2-1}\int_{s_0}^{\infty} s^{d/2-1}e^{-s}ds,$$

where $s_0 = \frac{d}{2e}t_0^{2/d}$. Define

$$\Gamma_{s_0}(d) = \int_{s_0}^{\infty} s^{d-1}e^{-s}ds.$$

(Note that when $s_0 = 0$ then $\Gamma_0(d)$ is the Gamma function.) Using integration by parts, we have

$$\Gamma_{s_0}(d) = s_0^{d-1}e^{-s_0} + (d-1)\Gamma_{s_0}(d-1). \tag{5}$$

Moreover, when $d \leq 1$, we have $\Gamma_{s_0}(d) \leq s_0^{d-1}\int_{s_0}^{\infty} e^{-s}ds = s_0^{d-1}e^{-s_0}$.

Note that if $t_0 \geq (2e)^{d/2}$, then $s_0 \geq d-2$. Hence, if we open the recursive definition of $\Gamma_{s_0}(d/2)$ in Equation (5), we have

$$\begin{aligned}
\Gamma_{s_0}(d/2) &\leq e^{-s_0}\sum_{i=0}^{\lceil\frac{d}{2}\rceil-1} s_0^{d/2-1-i}\prod_{j=1}^{i}(d/2-j)\\
&\leq e^{-s_0}s_0^{d/2-1}\sum_{i=0}^{\lceil\frac{d}{2}\rceil-1}\Big(\frac{d/2-1}{s_0}\Big)^i\\
&\leq 2e^{-s_0}s_0^{d/2-1},
\end{aligned}$$

because the summation is a geometric sum with ratio at most $1/2$. Substituting s_0 with t_0, we obtain

$$\begin{aligned}
e\Big(\frac{2e}{d}\Big)^{d/2-1}\int_{s_0}^{\infty} s^{d/2-1}e^{-s}ds &\leq 2e\Big(\frac{2e}{d}\Big)^{d/2-1}e^{-s_0}s_0^{d/2-1}\\
&= 2et_0^{1-2/d}e^{-\frac{d}{2e}t_0^{2/d}}.
\end{aligned}$$

◀

Proof of Claim 21. We rewrite $|f(x)|$ as $\int_0^{|f(x)|}\mathbb{1}dt = \int_0^{\infty}\mathbb{1}(|f(x)| \geq t)dt$ and obtain

$$\begin{aligned}
\mathop{\mathbb{E}}_{x\sim\{0,1\}^n}[g(x)|f(x)|] &= \mathop{\mathbb{E}}_{x\sim\{0,1\}^n}\Big[\int_0^{\infty} g(x)\mathbb{1}(|f(x)| \geq t)dt\Big]\\
&\leq \mathop{\mathbb{E}}_{x\sim\{0,1\}^n}\Big[\int_0^{\infty}\min\big\{g(x), \mathbb{1}(|f(x)| \geq t)\big\}dt\Big]\\
&= \int_0^{\infty}\min\Big\{\mathbb{E}[g], \Pr_x[|f(x)| \geq t]\Big\}dt\\
&\leq \int_0^{t_0}\mathbb{E}[g]dt + \int_{t_0}^{\infty}\Pr\big[|f(x)| \geq t\big]dt\\
&\leq \mathbb{E}[g]t_0 + \int_{t_0}^{\infty} e^{-\frac{d}{2e}t^{2/d}}dt.
\end{aligned}$$

Since $t_0 \geq (2e)^{d/2}$, by Claim 23 this is at most $\mathbb{E}[g]t_0 + 2et_0^{1-2/d}e^{-\frac{d}{2e}t_0^{2/d}}$. ◁

Proof of Lemma 10. Define f to be $f(x) := \sum_{|S|=d} \hat{f}_S \chi_S(x)$, where $\hat{f}_S = \hat{g}_S \big(\sum_{|T|=d} \hat{g}_T^2\big)^{-1/2}$. Note that $\|f\|_2 = 1$, and we have

$$\mathbb{E}[g(x)f(x)] = \frac{\sum_S \hat{g}_S \,\mathbb{E}[g(x)\chi_S(x)]}{\big(\sum_{|T|=d} \hat{g}_T^2\big)^{1/2}} = \Big(\sum_{|S|=d} \hat{g}_S^2\Big)^{1/2}.$$

Let $t_0 = (2e \ln(e/\,\mathbb{E}[g]^{1/d}))^{d/2} \geq (2e)^{d/2}$. By Claim 21,

$$\Big(\sum_{|S|=d} \hat{g}_S^2\Big)^{1/2} = \mathbb{E}[g(x)f(x)] \leq \mathbb{E}[g(x)|f(x)|] \leq \mathbb{E}[g]t_0 + 2et_0^{1-2/d} e^{-\frac{d}{2e}t_0^{2/d}}.$$

By our choice of t_0, the second term is at most

$$2et_0^{1-2/d} e^{-\frac{d}{2e}t_0^{2/d}} \leq \left(2e \ln\Big(\frac{e}{\mathbb{E}[g]^{1/d}}\Big)\right)^{d/2} \frac{\mathbb{E}[g]}{e^d} \leq (2/e)^{d/2}\,\mathbb{E}[g] \ln\Big(\frac{e}{\mathbb{E}[g]^{1/d}}\Big)^{d/2},$$

which is no greater than the first term. So

$$\Big(\sum_{|S|=d} \hat{g}_S^2\Big)^{1/2} \leq 2\,\mathbb{E}[g]\big(2e \ln(e/\,\mathbb{E}[g]^{1/d})\big)^{d/2},$$

and the lemma follows. ◀

References

1. Scott Aaronson. BQP and the polynomial hierarchy. In *42nd ACM Symp. on the Theory of Computing (STOC)*, pages 141–150. ACM, 2010.
2. Miklós Ajtai. Σ_1^1-formulae on finite structures. *Annals of Pure and Applied Logic*, 24(1):1–48, 1983.
3. Miklós Ajtai and Michael Ben-Or. A Theorem on Probabilistic Constant Depth Computations. In *16th ACM Symp. on the Theory of Computing (STOC)*, pages 471–474, 1984.
4. Miklós Ajtai, János Komlós, and Endre Szemerédi. Deterministic simulation in LOGSPACE. In *19th ACM Symp. on the Theory of Computing (STOC)*, pages 132–140, 1987.
5. Miklos Ajtai and Avi Wigderson. Deterministic simulation of probabilistic constant-depth circuits. *Advances in Computing Research - Randomness and Computation*, 5:199–223, 1989.
6. Kazuyuki Amano. Bounds on the Size of Small Depth Circuits for Approximating Majority. In *36th Coll. on Automata, Languages and Programming (ICALP)*, pages 59–70. Springer, 2009.
7. Roy Armoni, Michael E. Saks, Avi Wigderson, and Shiyu Zhou. Discrepancy Sets and Pseudorandom Generators for Combinatorial Rectangles. In *37th IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 412–421, 1996.
8. Andrej Bogdanov, Periklis A. Papakonstantinou, and Andrew Wan. Pseudorandomness for Read-Once Formulas. In *IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 240–246, 2011.
9. Joshua Brody and Elad Verbin. The Coin Problem, and Pseudorandomness for Branching Programs. In *51th IEEE Symp. on Foundations of Computer Science (FOCS)*, 2010.
10. Mei-Chu Chang. A polynomial bound in Freiman's theorem. *Duke Math. J.*, 113(3):399–419, 2002. doi:10.1215/S0012-7094-02-11331-3.
11. Suresh Chari, Pankaj Rohatgi, and Aravind Srinivasan. Improved algorithms via approximations of probability distributions. *J. Comput. System Sci.*, 61(1):81–107, 2000. doi:10.1006/jcss.1999.1695.

12 Eshan Chattopadhyay, Pooya Hatami, Kaave Hosseini, and Shachar Lovett. Pseudorandom Generators from Polarizing Random Walks. *Electronic Colloquium on Computational Complexity*, Technical Report TR18-015, 2018. URL: https://eccc.weizmann.ac.il/report/2018/015.

13 Eshan Chattopadhyay, Pooya Hatami, Shachar Lovett, and Avishay Tal. Pseudorandom generators from the second Fourier level and applications to AC0 with parity gates. In *ITCS'19—Proceedings of the 2019 ACM Conference on Innovations in Theoretical Computer Science*. ACM, 2019.

14 Eshan Chattopadhyay, Pooya Hatami, Omer Reingold, and Avishay Tal. Improved Pseudorandomness for Unordered Branching Programs through Local Monotonicity. In *ACM Symp. on the Theory of Computing (STOC)*, 2018.

15 Sitan Chen, Thomas Steinke, and Salil P. Vadhan. Pseudorandomness for Read-Once, Constant-Depth Circuits. *CoRR*, abs/1504.04675, 2015. arXiv:1504.04675.

16 Gil Cohen, Anat Ganor, and Ran Raz. Two Sides of the Coin Problem. In *Workshop on Randomization and Computation (RANDOM)*, pages 618–629, 2014. doi:10.4230/LIPIcs.APPROX-RANDOM.2014.618.

17 Anindya De, Omid Etesami, Luca Trevisan, and Madhur Tulsiani. Improved Pseudorandom Generators for Depth 2 Circuits. In *Workshop on Randomization and Computation (RANDOM)*, pages 504–517, 2010.

18 Irit Dinur, Ehud Friedgut, Guy Kindler, and Ryan O'Donnell. On the Fourier tails of bounded functions over the discrete cube. *Israel J. Math.*, 160:389–412, 2007. doi:10.1007/s11856-007-0068-9.

19 Dean Doron, Pooya Hatami, and William Hoza. Near-Optimal Pseudorandom Generators for Constant-Depth Read-Once Formulas, 2018. ECCC TR18-183.

20 Guy Even, Oded Goldreich, Michael Luby, Noam Nisan, and Boban Velickovic. Efficient approximation of product distributions. *Random Struct. Algorithms*, 13(1):1–16, 1998.

21 Michael A. Forbes and Zander Kelley. Pseudorandom Generators for Read-Once Branching Programs, in any Order. In *47th IEEE Symposium on Foundations of Computer Science (FOCS)*, 2018.

22 Parikshit Gopalan, Daniel Kane, and Raghu Meka. Pseudorandomness via the Discrete Fourier Transform. In *IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 903–922, 2015. doi:10.1109/FOCS.2015.60.

23 Parikshit Gopalan, Raghu Meka, Omer Reingold, Luca Trevisan, and Salil Vadhan. Better Pseudorandom Generators from Milder Pseudorandom Restrictions. In *IEEE Symp. on Foundations of Computer Science (FOCS)*, 2012.

24 Parikshit Gopalan, Rocco A. Servedio, and Avi Wigderson. Degree and sensitivity: tails of two distributions. In *31st Conference on Computational Complexity*, volume 50 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 13, 23. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2016.

25 Parikshit Gopalan and Amir Yehudayoff. Inequalities and tail bounds for elementary symmetric polynomials. *Electronic Colloquium on Computational Complexity (ECCC)*, 21:19, 2014. URL: http://eccc.hpi-web.de/report/2014/019.

26 Elad Haramaty, Chin Ho Lee, and Emanuele Viola. Bounded independence plus noise fools products. *SIAM J. Comput.*, 47(2):493–523, 2018. doi:10.1137/17M1129088.

27 Pooya Hatami and Avishay Tal. Pseudorandom generators for low-sensitivity functions. In *9th Innovations in Theoretical Computer Science*, volume 94 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 29, 13. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2018.

28 Russell Impagliazzo, Raghu Meka, and David Zuckerman. Pseudorandomness from Shrinkage. In *IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 111–119, 2012.

29 Russell Impagliazzo, Cristopher Moore, and Alexander Russell. An entropic proof of Chang's inequality. *SIAM J. Discrete Math.*, 28(1):173–176, 2014. doi:10.1137/120877982.

30 Russell Impagliazzo, Noam Nisan, and Avi Wigderson. Pseudorandomness for Network Algorithms. In *26th ACM Symp. on the Theory of Computing (STOC)*, pages 356–364, 1994.

31 Nathan Keller and Guy Kindler. Quantitative relation between noise sensitivity and influences. *Combinatorica*, 33(1):45–71, 2013. doi:10.1007/s00493-013-2719-2.

32 Swastik Kopparty and Srikanth Srinivasan. Certifying polynomials for $AC^0[\oplus]$ circuits, with applications to lower bounds and circuit compression. *Theory Comput.*, 14:Article 12, 24, 2018. doi:10.4086/toc.2018.v014a012.

33 Chin Ho Lee and Emanuele Viola. More on bounded independence plus noise: Pseudorandom generators for read-once polynomials. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 24, page 167, 2017.

34 Chin Ho Lee and Emanuele Viola. The coin problem for product tests. *ACM Trans. Comput. Theory*, 10(3):Art. 14, 10, 2018. doi:10.1145/3201787.

35 Nutan Limaye, Karteek Sreenivasiah, Srikanth Srinivasan, Utkarsh Tripathi, and S Venkitesh. The Coin Problem in Constant Depth: Sample Complexity and Parity gates. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume TR18–157, 2018.

36 Chi-Jen Lu. Improved pseudorandom generators for combinatorial rectangles. *Combinatorica*, 22(3):417–433, 2002.

37 Yishay Mansour. An $O(n^{\log\log n})$ learning algorithm for DNF under the uniform distribution. *J. Comput. System Sci.*, 50(3, part 3):543–550, 1995. Fifth Annual Workshop on Computational Learning Theory (COLT) (Pittsburgh, PA, 1992). doi:10.1006/jcss.1995.1043.

38 Raghu Meka, Omer Reingold, and Avishay Tal. Pseudorandom Generators for Width-3 Branching Programs. *arXiv preprint*, 2018. arXiv:1806.04256.

39 Joseph Naor and Moni Naor. Small-bias probability spaces: efficient constructions and applications. *SIAM J. on Computing*, 22(4):838–856, 1993.

40 Noam Nisan. Pseudorandom Generators for Space-bounded Computation. *Combinatorica*, 12(4):449–461, 1992.

41 Noam Nisan and David Zuckerman. Randomness is Linear in Space. *J. of Computer and System Sciences*, 52(1):43–52, February 1996.

42 Ryan O'Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2014.

43 Josip E. Pečarić, Frank Proschan, and Y. L. Tong. *Convex functions, partial orderings, and statistical applications*, volume 187 of *Mathematics in Science and Engineering*. Academic Press, Inc., Boston, MA, 1992.

44 Omer Reingold, Thomas Steinke, and Salil P. Vadhan. Pseudorandomness for Regular Branching Programs via Fourier Analysis. In *Workshop on Randomization and Computation (RANDOM)*, pages 655–670, 2013.

45 Benjamin Rossman and Srikanth Srinivasan. Separation of $AC^0[\oplus]$ formulas and circuits. In *44th International Colloquium on Automata, Languages, and Programming*, volume 80 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 50, 13. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2017.

46 Rocco A. Servedio and Li-Yang Tan. Improved pseudorandom generators from pseudorandom multi-switching lemmas. *CoRR*, abs/1801.03590, 2018. arXiv:1801.03590.

47 Ronen Shaltiel and Emanuele Viola. Hardness amplification proofs require majority. *SIAM J. on Computing*, 39(7):3122–3154, 2010.

48 J. Michael Steele. *The Cauchy-Schwarz master class*. MAA Problem Books Series. Mathematical Association of America, Washington, DC; Cambridge University Press, Cambridge, 2004. doi:10.1017/CBO9780511817106.

49 John P. Steinberger. The Distinguishability of Product Distributions by Read-Once Branching Programs. In *IEEE Conf. on Computational Complexity (CCC)*, pages 248–254, 2013. doi:10.1109/CCC.2013.33.

50 Thomas Steinke, Salil P. Vadhan, and Andrew Wan. Pseudorandomness and Fourier Growth Bounds for Width-3 Branching Programs. In *Workshop on Randomization and Computation (RANDOM)*, pages 885–899, 2014.

51 Avishay Tal. Tight Bounds on the Fourier Spectrum of AC0. In *Conf. on Computational Complexity (CCC)*, pages 15:1–15:31, 2017. doi:10.4230/LIPIcs.CCC.2017.15.

52 Michel Talagrand. How much are increasing sets positively correlated? *Combinatorica*, 16(2):243–258, 1996. doi:10.1007/BF01844850.

53 Luca Trevisan and TongKe Xue. A derandomized switching lemma and an improved derandomization of AC0. In *Computational Complexity (CCC), 2013 IEEE Conference on*, pages 242–247. IEEE, 2013.

54 Leslie G. Valiant. Short Monotone Formulae for the Majority Function. *J. Algorithms*, 5(3):363–366, 1984.

55 Emanuele Viola. On approximate majority and probabilistic time. *Computational Complexity*, 18(3):337–375, 2009.

56 Emanuele Viola. Randomness buys depth for approximate counting. *Computational Complexity*, 23(3):479–508, 2014.

57 Thomas Watson. Pseudorandom generators for combinatorial checkerboards. *Computational Complexity*, 22(4):727–769, 2013. doi:10.1007/s00037-012-0036-6.

Sherali–Adams Strikes Back

Ryan O'Donnell
Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, USA
https://www.cs.cmu.edu/~odonnell/
odonnell@cs.cmu.edu

Tselil Schramm
Harvard University, Cambridge, MA, USA
Massachusetts Institute of Technology, Cambridge, MA, USA
https://tselilschramm.org/
tselil@seas.harvard.edu

Abstract

Let G be any n-vertex graph whose random walk matrix has its nontrivial eigenvalues bounded in magnitude by $1/\sqrt{\Delta}$ (for example, a random graph G of average degree $\Theta(\Delta)$ typically has this property). We show that the $\exp\left(c\frac{\log n}{\log \Delta}\right)$-round Sherali–Adams linear programming hierarchy certifies that the maximum cut in such a G is at most 50.1% (in fact, at most $\frac{1}{2} + 2^{-\Omega(c)}$). For example, in random graphs with $n^{1.01}$ edges, $O(1)$ rounds suffice; in random graphs with $n \cdot \mathrm{polylog}(n)$ edges, $n^{O(1/\log\log n)} = n^{o(1)}$ rounds suffice.

Our results stand in contrast to the conventional beliefs that linear programming hierarchies perform poorly for MAX-CUT and other CSPs, and that eigenvalue/SDP methods are needed for effective refutation. Indeed, our results imply that constant-round Sherali–Adams can strongly refute random Boolean k-CSP instances with $n^{\lceil k/2 \rceil + \delta}$ constraints; previously this had only been done with spectral algorithms or the SOS SDP hierarchy.

2012 ACM Subject Classification Theory of computation → Linear programming; Theory of computation → Convex optimization; Theory of computation → Network optimization

Keywords and phrases Linear programming, Sherali–Adams, max-cut, graph eigenvalues, Sum-of-Squares

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.8

Related Version A version of the paper is available at https://arxiv.org/abs/1812.09967.

Funding *Ryan O'Donnell*: Supported by NSF grants CCF-1618679, CCF-1717606. This material is based upon work supported by the National Science Foundation under grant numbers listed above. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation (NSF).
Tselil Schramm: This work was supported by NSF grants CCF 1565264 and CNS 1618026.

Acknowledgements We thank Luca Trevisan for helpful comments, and Boaz Barak for suggesting the title. We also thank the Schloss Dagstuhl Leibniz Center for Informatics (and more specifically the organizers of the CSP Complexity and Approximability workshop), as well as the Casa Mathemática Oaxaca (and more specifically the organizers of the Analytic Techniques in TCS workshop); parts of this paper came together during discussions at these venues. T.S. also thanks the Oberwolfach Research Institute for Mathematics (and the organizers of the Proof Complexity and Beyond workshop), the Simons Institute (and the organizers of the Optimization semester program), and the Banff International Research Station (and the organizers of the Approximation Algorithms and Hardness workshop) where she tried to prove the opposite of the results in this paper, as well as Sam Hopkins, with whom some of those efforts were made.

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka; Article No. 8; pp. 8:1–8:30

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

1 Introduction

Linear programming (LP) is a fundamental algorithmic primitive, and is the method of choice for a huge number of optimization and approximation problems. Still, there are some very basic tasks where it performs poorly. A classic example is the simplest of all constraint satisfaction problems (CSPs), the MAX-CUT problem: Given a graph $G = (V, E)$, partition V into two parts so as to maximize the fraction of "cut" (crossing) edges. The standard LP relaxation for this problem [5, 38] involves optimizing over the *metric polytope*. Using "±1 notation", we have a variable Y_{uv} for each pair of vertices $\{u, v\}$ (with Y_{uv} supposed to be -1 if the edge is cut, $+1$ otherwise); the LP is:

$$\begin{aligned} \text{MAX-CUT}(G) \;\le\; \max\; & \frac{1}{2} - \frac{1}{2} \cdot \frac{1}{|E|} \sum_{uv \in E} \mathsf{Y}_{uv} & \\ & -1 \le \mathsf{Y}_{uv} \le 1 & \text{(for all } u, v \in V) \\ \text{s.t.} \quad & -\mathsf{Y}_{uv} - \mathsf{Y}_{vw} - \mathsf{Y}_{wu} \le 1 & \text{(for all } u, v, w \in V) \\ & -\mathsf{Y}_{uv} + \mathsf{Y}_{vw} + \mathsf{Y}_{wu} \le 1 & \text{(for all } u, v, w \in V) \end{aligned}$$

While this LP gives the optimal bound for some graphs (precisely, all graphs not contractible to K_5 [5]), it can give a very poor bound in general. Indeed, although there are graphs with maximum cut arbitrarily close to 1/2 (e.g., K_n), the above LP bound is at least 2/3 for every graph, since $\mathsf{Y}_{uv} \equiv -1/3$ is always a valid solution. Worse, there are graphs G with MAX-CUT(G) arbitrarily close to 1/2 but with LP value arbitrarily close to 1 – i.e., graphs where the *integrality ratio* is $2 - o(1)$. For example, this is true [39] of an Erdős–Rényi $\mathcal{G}(n, \Delta/n)$ random graph with high probability (whp) when $\Delta = \Delta(n)$ satisfies $\omega(1) < \Delta < n^{o(1)}$.

There have been two main strategies employed for overcoming this deficiency: strengthened LPs, and eigenvalue methods.

Strengthened LPs

One way to try to improve the performance of LPs on MAX-CUT is to add more valid inequalities to the LP relaxation, beyond just the "triangle inequalities". Innumerable valid inequalities have been considered: $(2k+1)$-gonal, hypermetric, negative type, gap, clique-web, suspended tree, as well as inequalities from the Lovász–Schrijver hierarchy; see Deza and Laurent [20, Ch. 28–30] for a review.

It is now known that the most principled and general form of this strategy is the *Sherali–Adams LP hierarchy* [45], reviewed in Section 2.4. At a high level, the Sherali–Adams LP hierarchy gives a standardized way to tighten LP relaxations of Boolean integer programs, by adding variables and constraints. The number of new variables/constraints is parameterized by a positive integer R, called the number of "rounds". Given a Boolean optimization problem with n variables, the R-round Sherali–Adams LP has variables and constraints corresponding to monomials of degree up to R, and thus has size $O(n)^R$. A remarkable recent line of work [13, 35] has shown that for any CSP (such as MAX-CUT), the R-round Sherali–Adams LP relaxation achieves essentially the tightest integrality ratio among all LPs of its size. Nevertheless, even this most powerful of LPs arguably struggles to certify good bounds for MAX-CUT. In a line of work [18, 44] concluding in a result of Charikar–Makarychev–Makarychev [15], it was demonstrated that for any constant $\epsilon > 0$, there are graphs (random Δ-regular ones, $\Delta = O(1)$) for which the $n^{\Omega(1)}$-round Sherali–Adams LP has a MAX-CUT integrality gap of $2 - \epsilon$. As a consequence, *every* MAX-CUT LP relaxation of size up to $2^{n^{\Omega(1)}}$ has such an integrality gap.

Eigenvalue and SDP methods

But for MAX-CUT, there is a simple, non-LP, algorithm that works very well to certify that random graphs have maximum cut close to 1/2: *eigenvalue* bounds. There are two slight variants here (that coincide in the case of regular graphs): Given graph $G = (V, E)$ with adjacency matrix A and diagonal degree matrix D, the eigenvalue bounds are

$$\text{MAX-CUT}(G) \leq \frac{|V|}{4|E|}\lambda_{\max}(D - A) \tag{1}$$

$$\text{MAX-CUT}(G) \leq \frac{1}{2} + \frac{1}{2}\lambda_{\max}(-D^{-1}A). \tag{2}$$

Here $D - A$ and $D^{-1}A$ are the *Laplacian* matrix and the *random walk* matrix, respectively. The use of eigenvalues to bound various cut values in graphs (problems like MAX-CUT, MIN-BISECTION, 2-XOR, expansion, etc.) has a long history dating back to Fieldler and Donath–Hoffman [25, 21] among others (Inequality (1) is specifically from Mohar–Poljak [37]). It was recognized early on that eigenvalue methods work particularly well for solving planted-random instances (e.g., of 2-XOR [32] and MIN-BISECTION [11]) and for certifying MAX-CUT values near 1/2 for truly random instances. Indeed, as soon as one knows (as we now do [46, 24]) that $D^{-1}A$ has all nontrivial eigenvalues bounded in magnitude by $O(1/\sqrt{\Delta})$ (whp) for a random Δ-regular graph (or an Erdős–Rényi $\mathcal{G}(n, \Delta/n)$ graph with $\Delta \gtrsim \log n$), the eigenvalue bound Inequality (2) certifies that $\text{MAX-CUT}(G) \leq 1/2 + O(1/\sqrt{\Delta})$. This implies an integrality ratio tending to 1; indeed, $\text{MAX-CUT}(G) = 1/2 + \Theta(1/\sqrt{\Delta})$ in such random graphs (whp).

Furthermore, if one extends the eigenvalue bound Inequality (1) above to

$$\text{MAX-CUT}(G) \leq \min_{\substack{U \text{ diagonal} \\ \text{tr}(U)=0}} \frac{|V|}{4|E|}\lambda_{\max}(D - A + U) \tag{3}$$

(as suggested by Delorme and Poljak [19], following [21, 11]), one obtains the polynomial-time computable *semidefinite programming (SDP)* bound. Goemans and Williamson [29] showed this bound has integrality ratio less than $1.14 \approx 1/.88$ for *worst-case* G, and it was subsequently shown [50, 23, 14] that the SDP bound is $1/2 + o(1)$ whenever $\text{MAX-CUT}(G) \leq 1/2 + o(1)$.

LPs cannot compete with eigenvalues/SDPs?

This seemingly striking separation between the performance of LPs and SDPs in the context of random MAX-CUT instances is now taken as a matter of course. To quote, e.g., [47],

> [E]xcept for semidefinite programming, we know of no technique that can provide, for every graph of max cut optimum $\leq .501$, a certificate that its optimum is $\leq .99$. Indeed, the results of [18, 44][1] show that large classes of Linear Programming relaxations of max cut are unable to distinguish such instances.

Specifically, the last statement here is true for Δ-regular random graphs when Δ is a certain large constant. The conventional wisdom is that for such graphs, linear programs cannot compete with semidefinite programs, and cannot certify even the eigenvalue bound.

Our main result challenges this conception.

[1] One would also add the subsequently written[15] here.

1.1 Our results

We show that whenever the eigenvalue bound Inequality (2) certifies the bound $\text{MAX-CUT}(G) \leq 1/2 + o(1)$, then $n^{o(1)}$-round Sherali–Adams can certify this as well.[2]

▶ **Theorem 1** (Simplified version of Theorem 30 and Corollary 31)**.** *Let G be a simple n-vertex graph and assume that $|\lambda| < \rho$ for all eigenvalues λ of G's random walk matrix $D^{-1}A$ (excluding the trivial eigenvalue of 1). Then for any $1 \leq c \leq \Omega(\log(1/\rho))$, Sherali–Adams with $n^{O(c/\log(1/\rho))}$ rounds certifies that $\text{MAX-CUT}(G) \leq 1/2 + 2^{-c}$.*

For example, if G's random walk matrix has its nontrivial eigenvalues bounded in magnitude by $n^{-.001}$, as is the case (whp) for random graphs with about $n^{1.002}$ edges, then Sherali–Adams can certify $\text{MAX-CUT}(G) \leq 50.1\%$ with *constantly* many rounds. We find this result surprising, and in defiance of the common belief that polynomial-sized LPs cannot take advantage of spectral properties of the underlying graph.

▶ Remark 2. We wish to emphasize that it is not the resulting $n^{O(1)}$ running time that is surprising; one can (and should) already achieve this with the eigenvalue bound. What is surprising is the inherent power of the Sherali–Adams relaxation itself.

▶ Remark 3. One might ask whether Theorem 1 even requires the assumption of small eigenvalues. That is, *perhaps $n^{o(1)}$-round Sherali–Adams can certify* $\text{MAX-CUT} \leq 1/2 + o(1)$ *whenever this is* true. We speculate that this may in fact be the case. As mentioned earlier, the basic SDP relaxation Inequality (3) has this property [50, 23, 14], meaning that whenever graph G has $\text{MAX-CUT}(G) \leq 1/2 + o(1)$, there is a traceless diagonal matrix U such that the eigenvalue bound applied to $A - U$ certifies the maxcut bound. It seems possible that our proof might be adapted to work with this $A - U$ rather than A, in which case Sherali–Adams would also have the property.

We add that the plain eigenvalue bound does *not* have this property: there exist graphs with large (nontrivial) eigenvalues even though the maximum cut is close to 1/2.[3]

1.1.1 Subexponential-sized LPs for max-cut in sparse random graphs

One setting in which the spectral radius ρ is understood concretely is in random regular graphs. Building upon [27, 12, 17], the following was recently shown:

▶ **Theorem** ([46])**.** *There is a fixed constant C such that for all $3 \leq \Delta \leq n/2$ with Δn even, it holds that a uniformly random n-vertex Δ-regular simple graph $\boldsymbol{G}$ satisfies the following with high probability: all eigenvalues of $\boldsymbol{G}$'s normalized adjacency matrix, other than 1, are at most $C/\sqrt{\Delta}$ in magnitude.*

Combining the above with Theorem 1, we have the following consequence for MAX-CUT on random regular graphs:

▶ **Corollary 4.** *Let n, $3 \leq \Delta \leq n/2$, and $1 \leq c \leq \Omega(\log \Delta)$ be positive integers. Then if G is a random Δ-regular n-vertex graph, with high probability $n^{O(c/\log \Delta)}$-round Sherali–Adams can certify that $\text{MAX-CUT}(G) \leq \frac{1}{2} + 2^{-c}$.*

---

[2] Actually, there is a slight mismatch between our result and Inequality (2): in Theorem 1 we need the maximum eigenvalue *in magnitude* to be small; i.e., we need $\lambda_{\min}(-D^{-1}A)$ to be not too negative. This may well just be an artifact of our proof.

[3] Consider, for example, a graph given by the union of a Δ-regular random graph on n vertices and a Δ-regular bipartite graph on $\sqrt{n}$ vertices. This will have MAX-CUT value close to 1/2, but will also have large negative eigenvalues coming from the bipartite component.

For example, if $\Delta \geq C \cdot 10^6$ (for C the constant in the bound on $\lambda(G)$), then $n^{1/3}$-rounds of Sherali–Adams can certify MAX-CUT$(G) \leq .51$. This result serves as a partial converse to [15]:

▶ **Theorem** ([15, Theorem 5.3]). *For every fixed integer $\Delta \geq 3$, with high probability over the choice of an n-vertex Δ-regular random graph G,*[4] *the $n^{\Theta(1/f(\Delta))}$-round Sherali–Adams relaxation for* MAX-CUT *has value at least* MAX-CUT$(G) \geq 0.99$*, where $f(\Delta)$ is a function that grows with Δ.*

While [15] show that Δ-regular random graphs require Sherali–Adams (and by [35], any LP) relaxations of at least subexponential size, our result implies that subexponential LPs are sufficient. Further, though the function $f(\Delta)$ is not specified in [15], by tracing back through citations (e.g. [3, 4, 16]) to extract a dependence, it appears we may take $f(\Delta) = \log \Delta$. So our upper bound is *tight* as a function of Δ, up to constant factors.

Prior to our result, it was unclear whether even $(n/\text{polylog}n)$-round Sherali–Adams could certify that the MAX-CUT value was bounded by .99 for sparse random regular graphs. Indeed, it was equally if not more conceivable that Charikar et al.'s result was not tight, and could be extended to $\tilde{\Omega}(n)$-rounds. In light of our result, we are left to wonder whether there are instances of MAX-CUT which have truly exponential extension complexity.

1.1.2 Refuting Random CSPs with linear programs

With minor modifications, our argument extends as well to 2-XOR. Then, following the framework in [2], we have the following consequence for certifying bounds on the value of random k-CSPs:

▶ **Theorem 5** (Simplified version of Theorem 38). *Suppose that $P : \{\pm 1\}^k \to \{0, 1\}$ is a k-ary Boolean predicate, and that $\delta, \epsilon > 0$. Let $\boldsymbol{E}[P]$ be the probability that a random $x \in \{\pm 1\}^k$ satisfies P. Then for a random instance $\mathcal{I}$ of P on n variables with $m \geq n^{\lceil k/2 \rceil + \delta}$ expected clauses, with high probability Sherali–Adams can certify that* $\text{OBJ}_{\mathcal{I}}(x) \leq \boldsymbol{E}[P] + \epsilon$ *using $R = O_{\epsilon,\delta,k}(1)$ rounds.*

This almost matches the comparable performance of Sum-of-Squares (SOS) and spectral algorithms [2], which are known to require $m \geq n^{k/2}$ clauses to certify comparable bounds in polynomial time [31, 43, 34].[5] Prior to our work it was known that Sherali–Adams admits *weak* refutations (i.e. a certificate that OBJ $\leq 1 - o(1)$) when $m \geq n^{k/2}$, but it was conceivable (and even conjectured) that $O(1)$-rounds could not certify OBJ $\leq 1 - \delta$ for constant δ at $m = o(n^k)$.

The result above also extends to t-wise independent predicates as in [2] (see Section 5). Also, one may extract the dependence on the parameters ϵ, δ to give nontrivial results when these parameters depend on n.[6]

---

[4] In [15], the graph is actually a *pruned* random graph, in which $o(n)$ edges are removed; this does not affect compatibility with our results, as the LP value is Lipschitz and so the pruning changes the LP value by $o(1)$.

[5] The expert may notice that we require the number of clauses $m \gg n^{\lceil k/2 \rceil}$, whereas the best Sum-of-Squares and spectral algorithms require only $m \gg n^{k/2}$. This is because we do not know how to relate the Sherali–Adams value of the objective function to its square (local versions of the Cauchy-Schwarz argument result in a loss). Such a relation would allow us to apply our techniques immediately to prove that Sherali–Adams matches the SOS and spectral performance for odd as well as even k.

[6] Though for 2-XOR and MAX-CUT we have done this explicitly, for higher-arity random CSPs we have left this for the interested reader.

1.2 Prior work

It is a folklore result that in random graphs with average degree n^δ, 3-round Sherali–Adams certifies a MAX-CUT value of at most $\max(1-\Omega(\delta), \frac{2}{3})$ (observed for the special case of $\delta > \frac{1}{2}$ in [6, 39]); this is simply because of concentration phenomena, since most edges participate in roughly the same number of odd cycles of length $O(\frac{1}{\delta}) \geq 3$, after which one can apply the triangle inequality. However this observation does not allow one to take the refutation strength independent of the average degree.

There are some prior works examining the performance of Sherali–Adams hierarchies on random (and otherwise "locally dense") CSPs. Building on works showing a PTAS for fully dense MAX-CUT [28], the work of de la Vega and Mathieu [18] shows that in graphs with average degree $\Omega(n)$, Sherali–Adams with $O(1)$ rounds certifies tight bounds on MAX-CUT. Subsequent works extended this to give a density/rounds tradeoff [49, 6]; the best of these shows that Sherali–Adams accurately estimates the MAX-CUT in graphs of average degree Δ using $O(n/\Delta)$ rounds. One may compare this to our theorem, which uses $n^{O(1/\log \Delta)}$ rounds for random graphs of average degree Δ.

Another relevant line of work is a series of LP hierarchy lower bounds (both for Sherali–Adams and for the weaker Lovász-Schrijver hierarchy) for problems such as MAX-CUT, VERTEX-COVER, and SPARSEST-CUT, including [1, 3, 18, 44], and culminating in the already mentioned result of Charikar, Makarychev and Makarychev; in [15], they give subexponential lower bounds on the number of rounds of Sherali–Adams required to strongly refute MAX-CUT in random regular graphs. Initially, one might expect that this result could be strengthened to prove that sparse random graphs require almost-exponential-sized LPs to refute MAX-CUT; our result demonstrates instead that [15] is almost tight.

We also mention the technique of *global correlation rounding* in the Sum-of-Squares hierarchy, which was used to give subexponential time algorithms for UNIQUE-GAMES [8] and polynomial-time approximations to MAX-BISECTION [42]. One philosophical similarity between these algorithms and ours is that both relate local properties (correlation among edges) to global properties (correlation of uniformly random pairs). But [8, 42] use the fact that the relaxation is an SDP (whereas our result is interesting *because* it is in the LP-only setting), and the "conditioning" steps that drive their algorithm are a fundamentally different approach.

There are many prior works concerned with certifying bounds on random CSPs, and we survey only some of them here, referring the interested reader to the discussion in [2]. The sequence of works [31, 43, 34] establishes Sum-of-Squares lower bounds for refuting any random constraint satisfaction problem, and these results are tight via the SOS algorithms of [2, 40]. The upshot is that for k-SAT and k-XOR,[7] $\omega(1)$ rounds of SOS are necessary to strongly refute an instance with $m = o(n^{k/2})$ clauses, and $O(1)$ rounds of SOS suffice when $m = \tilde{\Omega}(n^{k/2})$. Because SOS is a tighter relaxation than Sherali–Adams, the lower bounds [31, 43, 34] apply; our work can be seen to demonstrate that Sherali–Adams does not lag far behind SOS, strongly refuting with $O(1)$ rounds as soon as $m = \Omega(n^{\lceil k/2 \rceil + \delta})$ for any $\delta > 0$.

In a way, our result is part of a trend in anti-separation results for SDPs and simpler methods for pseudorandom and structured instances. For example, we have for planted clique that the SOS hierarchy performs no better than the Lovász-Schrijver+ hierarchy [22, 7], and also no better than a more primitive class of estimation methods based on local statistics (see

[7] This is more generally true for any CSP that supports a k-wise independent distribution over satisfying assignments.

e.g. [41] for a discussion). Similar results hold for problems relating to estimating the norms of random tensors [33]. Further, in [33] an equivalence is shown between SOS and spectral algorithms for a large class of average-case problems. Our result shows that for random CSPs, the guarantees of linear programs are surprisingly not far from the guarantees of SOS.

Finally, we mention related works in extended formulations. The sequence of works [13, 35] show that Sherali–Adams lower bounds for CSPs imply lower bounds for any LP relaxation; the stronger (and later) statement is due to [35], who show that subexponential-round integrality gaps for CSPs in the Sherali–Adams hierarchy imply subexponential-size lower bounds for any LP. These works are then applied in conjunction with [31, 43, 15] to give subexponential lower bounds against CSPs for any LP; our results give an upper limit to the mileage one can get from these lower bounds in the case of MAX-CUT, as we show that the specific construction of [15] cannot be strengthened much further.

1.3 Techniques

Our primary insight is that while Sherali–Adams is unable to reason about spectral properties *globally*, it does enforce that every set of R variables behave *locally* according to the marginals of a valid distribution, which induces *local* spectral constraints on every subset of up to R variables.

At first, it is unclear how one harnesses such local spectral constraints. But now suppose that we are in a graph whose adjacency matrix has a small spectral radius (excluding the trivial eeigenvalue). This implies that random walks mix rapidly, in say t steps, to a close-to-uniform distribution. Because a typical pair of vertices at distance t is distributed roughly as a uniformly random pair of vertices, any subset of R vertices which contains a path of length t already allows us to relate global and local graph properties.

To see why this helps, we take for a moment the "pseudoexpectation" view, in which we think of the R-round Sherali–Adams as providing a proxy for the degree-R moments of a distribution over MAX-CUT solutions $x \in \{\pm 1\}^n$, with MAX-CUT value

$$\text{MAX-CUT}(G) = \tfrac{1}{2} - \tfrac{1}{2} \mathop{\mathbf{E}}_{(u,v)\in E(G)} \widetilde{\mathbf{E}}[x_u x_v], \tag{4}$$

where $\widetilde{\mathbf{E}}[x_u x_v]$ is the "pseudo-correlation" of variables x_u, x_v. Because there is no globally consistent assignment, the pseudo-correlation $\widetilde{\mathbf{E}}[x_u x_v]$ for vertices u, v sampled uniformly at random will be close to 0.[8] But in any fixed subgraph of size $\Omega(t)$, enforcing $\widetilde{\mathbf{E}}[x_u x_v] \approx 0$ for pairs u, v at distance t has consequences, and limits the magnitude of correlation between pairs of adjacent vertices as well. In particular, because the pseudo-second moment matrix $\widetilde{\mathbf{E}}[x_S x_S^\top]$ for x_S the restriction of x to a set S of up to R vertices must be PSD, forcing some entries to 0 gives a constraint on the magnitude of edge correlations.

For example, suppose for a moment that we are in a graph G with $t = 2$, and that S is a star graph in G, given by one "root" vertex r with $k \le R - 1$ children $U = \{u_1, \dots, u_k\}$, and call $X = \widetilde{\mathbf{E}}[x_S x_S^\top] \succeq 0$. Notice that pairs of distinct children u_i, u_j are at distance $t = 2$ in S. If we then require $\widetilde{\mathbf{E}}[x_{u_i} x_{u_j}] = 0$ for every $u_i \neq u_j$, the only nonzero entries of X are the diagonals (which are all $\widetilde{\mathbf{E}}[x_i^2] = 1$), and the entries corresponding to edges from the root to its children, (r, u_i), which are $\widetilde{\mathbf{E}}[x_r x_{u_i}]$. Now defining the vector $c \in \mathbb{R}^S$ with a 1 at the root r, $c_r = 1$ and α on each child $u \in U$, $c_u = \alpha$, we have from the PSDness of X that

$$0 \le c^\top X c = \|c\|_2^2 + \sum_{u\in U} 2 c_r c_u \cdot \widetilde{\mathbf{E}}[x_r x_u] = (1 + \alpha^2 k) + 2\alpha k \mathop{\mathbf{E}}_{(u,v)\in E(S)} \widetilde{\mathbf{E}}[x_u x_v].$$

[8] This is implicit in our proof, but intuitively it should be true because e.g. u, v should be connected by equally many even- and odd-length paths.

Choosing $\alpha = k^{-1/2}$, this implies that within S, the average edge correlation is lower bounded by $\mathbf{E}_{(u,v)\in E[S]} \widetilde{\mathbf{E}}[x_u x_v] \geq -k^{-1/2}$. Of course, for a given star S we cannot know that $\widetilde{\mathbf{E}}[x_{u_i} x_{u_j}] = 0$, but if we take a well-chosen weighted average over *all* stars, this will (approximately) hold on average.

Our strategy is to take a carefully-chosen average over specific subgraphs S of G with $|S| = \Omega(t)$. By our choice of distribution and subgraph, the fact that the subgraphs *locally* have PSD pseudocorrelation matrices has consequences for the *global* average pseudocorrelation across edges, which in turn gives a bound on the objective value Equation (4). This allows us to show that Sherali–Adams certifies much better bounds than we previously thought possible, by aggregating local spectral information across many small subgraphs.

Organization

We begin with technical preliminaries in Section 2. In Section 3 we prove our main result. Section 4 establishes a mild lower bound for arbitrary graphs. Finally, Section 5 applies Theorem 1 to the refutation of arbitrary Boolean CSPs.

2 Setup and preliminaries

We begin by recalling preliminaries and introducing definitions that we will rely upon later.

2.1 Random walks on undirected graphs

Here, we recall some properties of random walks in undirected graphs that will be of use to us.

▶ **Definition 6.** *Let $G = (V, E)$ be an undirected finite graph, with parallel edges and self-loops allowed*[9]*, and with no isolated vertices. The* standard random walk on G *is the Markov chain on V in which at each step one follows a uniformly random edge out of the current vertex. For $u \in V$, we use the notation $\boldsymbol{v} \sim u$ to denote that $\boldsymbol{v}$ is the result of taking one random step from u.*

▶ **Definition 7.** *We write K for the transition operator of the standard random walk on G. That is, K is obtained from the adjacency matrix of G by normalizing the uth row by a factor of $1/\deg(u)$.*

▶ **Definition 8.** *We write π for the probability distribution on V defined by $\pi(v) = \frac{\deg(v)}{2|E|}$. As is well known, this is an invariant distribution for the standard random walk on G, and this Markov chain is reversible with respect to π. For $\boldsymbol{u} \sim \pi$ and $\boldsymbol{v} \sim \boldsymbol{u}$, the distribution of $(\boldsymbol{u}, \boldsymbol{v})$ is that of a uniformly random (directed) edge from E. We will also use the notation $\pi_* = \min_{v\in V}\{\pi(v)\}$.*

▶ **Definition 9.** *For $f, g : V \to \mathbb{R}$ we use the notation $\langle f, g\rangle_\pi$ for $\boldsymbol{E}_{\boldsymbol{u}\sim\pi}[f(\boldsymbol{u})g(\boldsymbol{u})]$. This is an inner product on the vector space $\mathbb{R}^V$; in case G is regular and hence π is the uniform distribution, it is the usual inner product scaled by a factor of $1/|V|$. It holds that*

$$\langle f, Kg\rangle_\pi = \langle Kf, g\rangle_\pi = \boldsymbol{E}_{(\boldsymbol{u},\boldsymbol{v})\sim E}[f(\boldsymbol{u})g(\boldsymbol{v})]. \tag{5}$$

[9] Self-loops count as "half an edge", and contribute 1 to a vertex's degree.

▶ **Definition 10.** *A* stationary d-step walk *is defined to be a sequence* $(\boldsymbol{u}_0, \boldsymbol{u}_1, \dots, \boldsymbol{u}_d)$ *formed by choosing an initial vertex* $\boldsymbol{u}_0 \sim \pi$*, and then taking a standard random walk, with* $\boldsymbol{u}_t \sim \boldsymbol{u}_{t-1}$*. Generalizing Equation* (5)*, it holds in this case that*

$$\boldsymbol{E}[f(\boldsymbol{u}_0)g(\boldsymbol{u}_d)] = \langle f, K^d g\rangle_\pi.$$

2.2 Tree-indexed random walks

To prove our main theorem we define a class of homomorphisms we call *tree-indexed random walks*.

▶ **Definition 11.** *Suppose we have a finite undirected tree with vertex set* T*. A* stationary T-indexed random walk in G *is a random homomorphism* $\boldsymbol{\phi} : T \to V$ *defined as follows: First, root the tree at an arbitrary vertex* $i_0 \in T$*. Next, define* $\boldsymbol{\phi}(i_0) \sim \pi$*. Then, independently for each "child"* j *of* i_0 *in the tree, define* $\boldsymbol{\phi}(j) \sim \boldsymbol{\phi}(i_0)$*; that is, define* $\boldsymbol{\phi}(j) \in V$ *to be the result of taking a random walk step from* $\boldsymbol{\phi}(i_0)$*. Recursively repeat this process for all children of* i_0*'s children, etc., until each vertex* $k \in T$ *has been assigned a vertex* $\boldsymbol{\phi}(k) \in V$*.*

We note that the homomorphism $\boldsymbol{\phi}$ defining the T-indexed random walk need not be injective. Consequently, if T is a tree with maximum degree D, we can still have a T-indexed random walk in a d-regular graph with $d < D$.

The following fact is simple to prove; see, e.g., [36].

▶ **Fact 12.** *The definition of* $\boldsymbol{\phi}$ *does not depend on the initially selected root* $i_0 \in T$*. Further, for any two vertices* $i, j \in T$ *at tree-distance* d*, if* $i = i_0, i_1, \dots, i_d = j$ *is the unique path in the tree between them, then the sequence* $(\boldsymbol{\phi}(i_0), \boldsymbol{\phi}(i_1), \dots, \boldsymbol{\phi}(i_d))$ *is distributed as a stationary* d*-step walk in* G*.*

2.3 2XOR and signed random walks

The 2-XOR constraint satisfaction problem is defined by instances of linear equations in $\mathbb{F}_2^n$. For us it will be convenient to associate with these instances a graph with signed edges, and on such graphs we perform a slightly modified random walk.

▶ **Definition 13.** *We assume that for each vertex pair* (u, v) *where* G *has edge, there is an associated* sign $\xi_{uv} = \xi_{vu} \in \{\pm 1\}$.[10] *We arrange these signs into a symmetric matrix* $\Xi = (\xi_{uv})_{uv}$*. If* G *has no* (u, v) *edge then the entry* Ξ_{uv} *will not matter; we can take it to be* 0*.*

▶ **Definition 14.** *We write* $\overline{K} = \Xi \circ K$ *for the* signed transition operator*. The operator* $\overline{K}$ *is self-adjoint with respect to* $\langle \cdot, \cdot \rangle_\pi$*, and hence has real eigenvalues. It also holds that*

$$\langle f, \overline{K} g\rangle_\pi = \langle \overline{K} f, g\rangle_\pi = \mathop{\boldsymbol{E}}_{(\boldsymbol{u},\boldsymbol{v})\sim E}[\xi_{\boldsymbol{uv}} f(\boldsymbol{u}) g(\boldsymbol{v})]. \tag{6}$$

▶ **Definition 15.** *We may think of* G *and* Ξ *as defining a* 2-XOR *constraint satisfaction problem (CSP), in which the task is to find a labeling* $f : V \to \{\pm 1\}$ *so as to maximize the fraction of edges* $(u, v) \in E$ *for which the constraint* $f(u)f(v) = \xi_{uv}$ *is satisfied. The fraction of satisfied constraints is*

$$\mathop{\boldsymbol{E}}_{(\boldsymbol{u},\boldsymbol{v})\sim E}\left[\tfrac{1}{2} + \tfrac{1}{2}\xi_{\boldsymbol{uv}} f(\boldsymbol{u}) f(\boldsymbol{v})\right] = \tfrac{1}{2} + \tfrac{1}{2}\langle f, \overline{K} f\rangle_\pi. \tag{7}$$

We will typically ignore the $\frac{1}{2}$*'s and think of the* 2-XOR *CSP as maximizing the quadratic form* $\langle f, \overline{K} f\rangle_\pi$*. When all signs in the matrix* Ξ *are* -1*, we refer to this as the* MAX-CUT *CSP.*

[10] If G has multiple (u, v) edges, we think of them as all having the same sign.

▶ **Definition 16.** *We say that a* signed *stationary d-step walk is a sequence of pairs* $(\boldsymbol{u}_t, \boldsymbol{\sigma}_t) \in \{\pm 1\} \times V$ *for* $0 \leq t \leq d$*, chosen as follows: first, we choose a stationary d-step walk* $(\boldsymbol{u}_0, \boldsymbol{u}_1, \ldots, \boldsymbol{u}_d)$ *in G; second, we choose* $\boldsymbol{\sigma}_0 \in \{\pm 1\}$ *uniformly at random; finally, we define* $\boldsymbol{\sigma}_t = \boldsymbol{\sigma}_{t-1}\xi_{\boldsymbol{\sigma}_{t-1}\boldsymbol{\sigma}_t}$*. Generalizing Equation* (6)*, it holds in this case that*

$$\boldsymbol{E}[\boldsymbol{\sigma}_0 f(\boldsymbol{u}_0)\boldsymbol{\sigma}_d g(\boldsymbol{u}_d)] = \langle f, \overline{K}^d g\rangle_\pi.$$

▶ **Definition 17.** *We extend the notion from Definition 11 to that of a* signed *stationary T-indexed random walk in G. Together with the random homomorphism* $\boldsymbol{\phi} : T \to V$*, we also choose a random signing* $\boldsymbol{\sigma} : T \to \{\pm 1\}$ *as follows: for the root* i_0*, the sign* $\boldsymbol{\sigma}(i_0) \in \{\pm 1\}$ *is chosen uniformly at random; then, all other signs are deterministically chosen – for each j of* i_0 *we set* $\boldsymbol{\sigma}(j) = \xi_{i_0 j}\boldsymbol{\sigma}(i_0)$*, and in general* $\boldsymbol{\sigma}(k) = \xi_{k'k}\boldsymbol{\sigma}(k)$ *where* k' *is the parent of k. Again, it is not hard to show that the definition of* $(\boldsymbol{\phi}, \boldsymbol{\sigma})$ *does not depend on the choice of root* i_0*, and that for any path* $i_0, i_1, \ldots, i_d$ *of vertices in the tree, the distribution of* $(\boldsymbol{\phi}(i_0), \boldsymbol{\sigma}(i_0)), (\boldsymbol{\phi}(i_1), \boldsymbol{\sigma}(i_1)), \ldots (\boldsymbol{\phi}(i_d), \boldsymbol{\sigma}(i_d))$ *is that of a signed stationary d-step walk in G.*

2.4 Proof systems

Our central object of study is the Sherali–Adams proof system, although our results also apply to a weaker proof system (see Remark 24). We first define Sherali–Adams in this "proof system" format (as opposed to the original optimization format); see, e.g., [9] for some commentary on this perspective.

▶ **Definition 18.** *Let* $\mathsf{X}_1, \ldots, \mathsf{X}_n$ *be indeterminates that are supposed to stand for real numbers* ± 1*. The R-*round Sherali–Adams proof system *[45] may be defined as follows: The "lines" of the proof are real polynomial inequalities in* $\mathsf{X}_1, \ldots, \mathsf{X}_n$ *(where the polynomials may as well be multilinear). The allowed "axioms" are any real inequalities of the form* $q(\mathsf{X}_{u_1}, \ldots, \mathsf{X}_{u_R}) \geq 0$*, where the inequality is* true *for every* ± 1 *assignment to the indeterminates* X_{u_i}*. The "deduction rules" allow one to derive any nonnegative linear combination of previous lines. This is a sound proof system for inequalities about* ± 1 *numbers* $\mathsf{X}_1, \ldots, \mathsf{X}_n$*.*

▶ **Fact 19.** *There is a* $\mathrm{poly}(n^R, L)$*-time algorithm based on Linear Programming for determining whether a given polynomial inequality* $p(\mathsf{X}) \geq 0$ *of degree at most R (and rational coefficients of total bit-complexity L) is derivable in the R-round Sherali–Adams proof system.*

As mentioned earlier, it will helpful for us to take a "Sum-of-Squares" perspective on Sherali–Adams. The well-known fact here is that a multilinear polynomial $q(\mathsf{X}_{u_1}, \ldots, \mathsf{X}_{u_R})$ is nonnegative for all ± 1 assignments if and only if it can be represented as the (multilinearization of) a squared polynomial p^2 on R indeterminates. (This p will be the unique "Fourier expansion" for the function $\sqrt{q} : \{\pm 1\}^R \to \mathbb{R}$; again, see [9] for some discussion.) Let us now define a proof system that can encapsulate both Sherali–Adams and SOS:

▶ **Definition 20.** *We define the R-*local, degree-D (static) Sum-of-Squares (SOS) proof system *over indeterminates* $\mathsf{X}_1, \ldots, \mathsf{X}_n$ *as follows. The "lines" of the proof are real polynomial inequalities in* $\mathsf{X} = (\mathsf{X}_1, \ldots, \mathsf{X}_n)$*. The default "axioms" are any real inequalities of the form* $p(\mathsf{X}_{u_1}, \ldots, \mathsf{X}_{u_R})^2 \geq 0$*, where p is a polynomial in at most R variables and of degree at most* $D/2$*. The "deduction rules" allow one to derive any nonnegative linear combination of previous lines. This is a sound proof system for inequalities about n real numbers* $\mathsf{X}_1, \ldots, \mathsf{X}_n$*.*

In addition to the default axioms, one may also sometimes include problem-specific "equalities" of the form $q(\mathsf{X}) = 0$*. In this case, one is allowed additional axioms of the form* $q(\mathsf{X})s(\mathsf{X}) \gtrless 0$ *the polynomial* $q(\mathsf{X})s(\mathsf{X})$ *depends on at most R indeterminates and has degree at most D.*

▶ **Fact 21.** *The case of $R = \infty$ (equivalently, $R = n$) corresponds to the well-known* degree-D SOS proof system.

▶ **Definition 22.** *Suppose one includes the* Boolean equalities, *meaning $\mathsf{X}_u^2 - 1 = 0$ for all $1 \le i \le n$.*[11] *In this case $D = \infty$ is equivalent to $D = R$, and the corresponding proof system is* equivalent *to R-round Sherali–Adams.*

▶ **Fact 23.** *We will often be concerned with the R-local, degree-2 SOS proof system, where all lines are quadratic inequalities. In this case, we could equivalently state that the default axioms are all those inequalities of the form*

$$\mathsf{x}^\top P \mathsf{x} \ge 0, \tag{8}$$

where $\mathsf{x} = (\mathsf{X}_{u_1}, \dots, \mathsf{X}_{u_R})$ is a length-R subvector of X, and P is an $R \times R$ positive semidefinite (PSD) matrix.

▶ Remark 24. In fact, we will often be concerned with the R-round, degree-2 Sherali–Adams proof system, which is strictly weaker than the general R-round Sherali–Adams proof system. Despite this restriction to $D = 2$, we only know the poly(n^R, L)-time algorithm for deciding derivability of a given quadratic polynomial $p(\mathsf{X}) \ge 0$ (of bit-complexity L).

3 Proof of Main Theorem (2XOR certifications from "spider walks")

In this section, we prove our main theorem: given a 2-XOR or MAX-CUT instance on a graph G with small spectral radius, we will show that the R-local degree-2 SOS proof system gives nontrivial refutations with R not too large.

Our strategy is as follows: we select a specific tree T of size $\propto R$, and we consider the distribution over copies of T in our graph given by the T-indexed stationary random walk. We will use this distribution to define the coefficients for a degree-2, R-local proof that bounds the objective value of the CSP. We will do this by exploiting the uniformity of the graph guaranteed by the small spectral radius, and the fact that degree-2 R-local SOS proofs can certify positivity of quadratic forms $c^\top \mathsf{X}|_S \mathsf{X}|_S^\top c$, where $X|_S$ is the restriction of X to a set S of variables with $|S| \le R$ and $c \in \mathbb{R}^{|S|}$.

Intuitively, in the "pseudoexpectation" view, the idea of our proof is as follows. When there is no globally consistent assignment, a uniformly random pair of vertices $u, v \in V$ will have pseudocorrelation close to zero. On the other hand, if t-step random walks mix to a roughly uniform distribution over vertices in the graph, then pairs of vertices at distance t will also have pseudocorrelation close to zero. But also, in our proof system the degree-2 pseudomoments of up to R variables obey a positive-semidefiniteness constraint. By choosing the tree T with diameter at least t, while also choosing T to propagate the effect of the low-pseudocorrelation at the diameter to give low-pseudocorrelation on signed edges, we show that the proof system can certify that the objective value is small. Specifically, we will choose T to be a *spider graph*:

▶ **Definition 25.** *For integers $k, \ell \in \mathbb{N}^+$, we define a* (k, ℓ)-spider graph *to be the tree formed by gluing together k paths of length ℓ at a common endpoint called the* root. *This spider has $k\ell + 1$ vertices and diameter 2ℓ.*

[11] Or alternatively, $\mathsf{X}_u^2 - \mathsf{X}_u = 0$ for all i.

While we were not able to formally prove that the spider is the optimal choice of tree, intuitively, we want to choose a tree that maximizes the ratio of the number of pairs at maximum distance (since such pairs relate the local properties to the global structure) to the number of vertices in the tree (because we need to take our number of rounds R to be at least the size of the tree). Among trees, the spider is the graph that maximizes this ratio.

Let us henceforth fix a (k, ℓ)-spider graph, where the parameters k and ℓ will be chosen later. We write S for the vertex set of this tree (and sometimes identify S with the tree itself).

▶ **Definition 26.** *For $0 \le d \le 2\ell$, we define the matrix $A^{(d)} \in \mathbb{R}^{S \times S}$ to be the "distance-d" adjacency matrix of the spider; i.e., $A^{(d)}_{ij}$ is 1 if $\mathrm{dist}_S(i,j) = d$ and is 0 otherwise. (We remark that $A^{(0)}$ is the identity matrix.)*

The following key technical theorem establishes the existence of a matrix Ψ which will allow us to define the coefficients in our R-local degree-2 SOS proof. It will be proven in Section 3.2:

▶ **Theorem 27.** *For any parameter $\alpha \in R$, there is a PSD matrix $\Psi = \Psi_\alpha \in \mathbb{R}^{S \times S}$ with the following properties:*

$$\begin{aligned}
\langle \Psi, A^{(0)} \rangle &= 1 + \frac{1}{2k}\alpha^{2\ell} + \frac{1}{k-1}\frac{\alpha^{2\ell} - \alpha^2}{\alpha^2 - 1}, \\
\langle \Psi, A^{(1)} \rangle &= \alpha, \\
\langle \Psi, A^{(d)} \rangle &= 0 \quad \textit{for } 1 < d < 2\ell, \\
\langle \Psi, A^{(2\ell)} \rangle &= \frac{1 - 1/k}{2}\alpha^{2\ell}.
\end{aligned}$$

Here we are using the notation $\langle B, C \rangle$ for the "matrix (Frobenius) inner product" $\mathrm{Tr}(B^\top C)$.

▶ **Corollary 28.** *Assuming that $k \ge 3^\ell$ and taking $\alpha = k^{1/(2\ell)}$, the PSD matrix Ψ satisfies the following four statements:*

$$\begin{aligned}
&3/2 \le \langle \Psi, A^{(0)} \rangle \le 2, && \langle \Psi, A^{(1)} \rangle = k^{1/(2\ell)}, \\
&\langle \Psi, A^{(d)} \rangle = 0 \textit{ for } 1 < d < 2\ell, && \langle \Psi, A^{(2\ell)} \rangle = \tfrac{1}{2}(k-1).
\end{aligned}$$

We will also use the following small technical lemma:

▶ **Lemma 29.** *Let $M \in \mathbb{R}^{V \times V}$ and recall $\pi_* = \min_{v \in V}\{\pi(v)\} > 0$. Then the 2-local, degree-2 SOS proof system can derive*

$$\underset{\boldsymbol{u} \sim \pi}{\boldsymbol{E}} \sum_{v \in V} M_{\boldsymbol{u}v} \mathsf{X}_{\boldsymbol{u}} \mathsf{X}_v \le \pi_*^{-1/2} \|M\|_2 \underset{\boldsymbol{u} \sim \pi}{\boldsymbol{E}} \mathsf{X}^2_{\boldsymbol{u}}.$$

Proof. The proof system can derive the following inequality for any $\gamma > 0$, since the difference of the two sides is a perfect square:

$$M_{uv}\mathsf{X}_u\mathsf{X}_v \le \frac{M^2_{uv}}{2\gamma\pi(v)}\mathsf{X}^2_u + \frac{\gamma\pi(v)}{2}\mathsf{X}^2_v.$$

Thus it can derive

$$\underset{\boldsymbol{u} \sim \pi}{\mathbf{E}} \sum_{v \in V} M_{\boldsymbol{u}v} \mathsf{X}_{\boldsymbol{u}} \mathsf{X}_v \le \underset{\boldsymbol{u} \sim \pi}{\mathbf{E}} \mathsf{X}^2_{\boldsymbol{u}} \sum_{v \in V} \frac{M^2_{\boldsymbol{u}v}}{2\gamma\pi(v)} + \frac{\gamma}{2} \underset{\boldsymbol{v} \sim \pi}{\mathbf{E}} \mathsf{X}^2_v. \tag{9}$$

We'll take $\gamma = \pi_*^{-1/2}\|M\|_2$. Since we can certainly derive $a\mathsf{X}^2_u \le b\mathsf{X}^2_u$ whenever $a \le b$, we see that it suffices to establish

$$\sum_{v \in V} \frac{M^2_{\boldsymbol{u}v}}{2\gamma\pi(v)} \le \frac{\gamma}{2}$$

for every outcome of $\boldsymbol{u}$. But this is implied by $\sum_v M_{\boldsymbol{u}v}^2 \leq (\pi(v)/\pi_*)\|M\|_2^2$ for all $v \in V$, which is indeed true. ◀

We can now prove the following main theorem:

▶ **Theorem 30.** *Given parameters $k \geq 3^\ell$, let $R = k\ell + 1$ and define*

$$\beta = \frac{k\pi_*^{-1/2}}{2k^{1/(2\ell)}}\rho(\overline{K})^{2\ell} + \frac{2}{k^{1/(2\ell)}},$$

where $\rho(\overline{K})$ denotes the spectral radius of the signed transition matrix $\overline{K}$. Then R-local, degree-2 SOS can deduce the bound "$\rho(\overline{K}) \leq \beta$"; more precisely, it can deduce the two inequalities

$$-\beta\langle \mathsf{X}, \mathsf{X}\rangle_\pi \leq \langle \mathsf{X}, \overline{K}\mathsf{X}\rangle_\pi \leq \beta\langle \mathsf{X}, \mathsf{X}\rangle_\pi.$$

Before proving this theorem, let us simplify the parameters. For any $\epsilon > 0$, we can choose ℓ to be the smallest integer so that $(\frac{1}{\epsilon}\rho(\overline{K}))^{2\ell}\pi_*^{-1/2} \leq \epsilon$, and $k = \lceil(\frac{1}{\epsilon})^{2\ell}\rceil$. This gives the corollary:

▶ **Corollary 31.** *Suppose we have a graph $G = (V, E)$ with signed transition operator $\overline{K}$ and $\pi_* = \min_{v\in V}\frac{\deg(v)}{2|E|}$. Given $\epsilon > \min(\pi_*^{-1/2}, \rho(\overline{K}))$, take $\ell = \left\lceil \frac{1}{4}\frac{\log(\epsilon^2\pi_*)}{\log(\rho(\overline{K})/\epsilon)}\right\rceil$, and take $k = \lceil(\frac{1}{\epsilon})^{2\ell}\rceil$. Then for $R = k\ell + 1$, it holds that R-local degree-2 SOS can deduce the bound $\rho(\overline{K}) \leq \frac{5}{2}\epsilon$. In particular, if we think of G, Ξ as a* 2-XOR *CSP, it holds that R-round Sherali–Adams can deduce the bound* $\mathrm{OBJ} \leq \frac{1}{2} + \frac{5}{4}\epsilon$.

Proof. Taking the parameters as above, and using that the constraints $\mathsf{X}_u^2 = 1$ imply that R-round Sherali–Adams can deduce that $\langle \mathsf{X}, \mathsf{X}\rangle_\pi = 1$ whenever $R \geq 2$, and that as noted in Equation (7), $\mathrm{OBJ}(\mathsf{X}) = \frac{1}{2} + \frac{1}{2}\langle \mathsf{X}, \overline{K}\mathsf{X}\rangle_\pi$, so Theorem 30 gives the result. ◀

Corollary 31 implies the 2-XOR version of Theorem 1 since in simple graphs, $\log\frac{1}{\pi^*} = \Theta(\log n)$.

Proof of Theorem 30. For our (k, ℓ)-spider graph on S, let $(\boldsymbol{\phi}, \boldsymbol{\sigma})$ be a signed stationary S-indexed random walk in G. Define $\overline{\mathbf{x}}$ to be the S-indexed vector with $\overline{\mathbf{x}}_i = \boldsymbol{\sigma}(i)\mathsf{X}_{\boldsymbol{\phi}(i)}$. Then letting Ψ be the PSD matrix from Corollary 28, the R-local, degree-2 SOS proof system can derive

$$\langle \Psi, \overline{\mathbf{x}}\overline{\mathbf{x}}^\top\rangle = \overline{\mathbf{x}}^\top\Psi\overline{\mathbf{x}} \geq 0.$$

(This is in the form of Inequality (8) if we take $P = \mathrm{diag}(\boldsymbol{\sigma})\Psi\,\mathrm{diag}(\boldsymbol{\sigma})$.) Furthermore, the proof system can deduce this inequality in expectation; namely,

$$\langle \Psi, \mathsf{Y}\rangle \geq 0, \text{ where } \mathsf{Y} = \mathbf{E}[\overline{\mathbf{x}}\overline{\mathbf{x}}^\top]. \tag{10}$$

Now by the discussion in Definitions 16 and 17,

$$\mathsf{Y}_{ij} = \mathbf{E}[\boldsymbol{\sigma}(i)\mathsf{X}_{\boldsymbol{\phi}(i)}\boldsymbol{\sigma}(j)\mathsf{X}_{\boldsymbol{\phi}(j)}] = \langle \mathsf{X}, \overline{K}^{\mathrm{dist}_S(i,j)}\mathsf{X}\rangle_\pi. \tag{11}$$

Thus recalling the notation $A^{(d)}$ from Definition 26,

$$\mathsf{Y} = \sum_{d=0}^{2\ell}\langle \mathsf{X}, \overline{K}^d\mathsf{X}\rangle_\pi A^{(d)}, \tag{12}$$

and hence from Inequality (10) we get that R-local, degree-2 SOS can deduce

$$0 \le \sum_{d=0}^{2\ell} \langle \Psi, A^{(d)} \rangle \langle \mathsf{X}, \overline{K}^d \mathsf{X} \rangle_\pi = c_0 \langle \mathsf{X}, \mathsf{X} \rangle_\pi + k^{1/(2\ell)} \langle \mathsf{X}, \overline{K} \mathsf{X} \rangle_\pi + \tfrac{1}{2}(k-1) \langle \mathsf{X}, \overline{K}^{2\ell} \mathsf{X} \rangle_\pi, \tag{13}$$

for some constant $3/2 \le c_0 \le 2$ (here we used Corollary 28). Regarding the last term, we have:

$$\langle \mathsf{X}, \overline{K}^{2\ell} \mathsf{X} \rangle_\pi = \mathop{\mathbf{E}}_{\boldsymbol{u} \sim \pi} \sum_{v \in V} (\overline{K}^{2\ell})_{\boldsymbol{u}v} \mathsf{X}_{\boldsymbol{u}} \mathsf{X}_v. \tag{14}$$

If we cared only about the Sherali–Adams proof system with Boolean equalities, we would simply now deduce

$$\begin{aligned} \mathop{\mathbf{E}}_{\boldsymbol{u} \sim \pi} \sum_{v \in V} (\overline{K}^{2\ell})_{\boldsymbol{u}v} \mathsf{X}_{\boldsymbol{u}} \mathsf{X}_v &\le \mathop{\mathbf{E}}_{\boldsymbol{u} \sim \pi} \sum_{v \in V} \left| (\overline{K}^{2\ell})_{\boldsymbol{u}v} \right| \\ &\le \sqrt{|V|} \mathop{\mathbf{E}}_{\boldsymbol{u} \sim \pi} \|\overline{K}^{2\ell}_{\boldsymbol{u},\cdot}\|_2 \le \sqrt{|V|} \rho(\overline{K}^{2\ell}) = \sqrt{|V|} \rho(\overline{K})^{2\ell}, \end{aligned}$$

and later combine this with $c_0 \langle \mathsf{X}, \mathsf{X} \rangle_\pi = c_0$. But proceeding more generally, we instead use Lemma 29 to show that our proof system can derive

$$\mathop{\mathbf{E}}_{\boldsymbol{u} \sim \pi} \sum_{v \in V} (\overline{K}^{2\ell})_{\boldsymbol{u}v} \mathsf{X}_{\boldsymbol{u}} \mathsf{X}_v \le \pi_*^{-1/2} \rho(\overline{K})^{2\ell} \langle \mathsf{X}, \mathsf{X} \rangle_\pi.$$

Putting this into Equation (14) and Inequality (13) we get

$$\langle \mathsf{X}, \overline{K} \mathsf{X} \rangle_\pi \ge -\frac{c_0 + \frac{1}{2}(k-1)\pi_*^{-1/2} \rho(\overline{K})^{2\ell}}{k^{1/(2\ell)}} \langle \mathsf{X}, \mathsf{X} \rangle_\pi \ge -\beta \langle \mathsf{X}, \mathsf{X} \rangle_\pi.$$

Repeating the derivation with $-\overline{K}$ in place of $\overline{K}$ completes the proof. ◀

3.1 Max-Cut

The following theorem is quite similar to Theorem 30. In it, we allow K to have the large eigenvalue 1, and only certify that it has no large-magnitude negative eigenvalue. The subsequent corollary is deduced identically to Corollary 31.

▶ **Theorem 32.** *Given transition operator K for the standard random walk on G, let $K' = K - J$, where J is the all-1's matrix. For parameters $k \ge 3^\ell$, let $R = k\ell + 1$ and define*

$$\beta = \frac{k\pi_*^{-1/2}}{2k^{1/(2\ell)}} \rho(K')^{2\ell} + \frac{2}{k^{1/(2\ell)}}.$$

(Note that $\rho(K')$ is equal to maximum-magnitude eigenvalue of K when the trivial 1 eigenvalue is excluded.) Then $2R$-local, degree-2 SOS can deduce the bound "$\lambda_{\min}(K) \ge -\beta$"; more precisely, it can deduce the inequality

$$\langle \mathsf{X}, \overline{K} \mathsf{X} \rangle_\pi \ge -\beta \langle \mathsf{X}, \mathsf{X} \rangle_\pi.$$

▶ **Corollary 33.** *Suppose we have a graph $G = (V, E)$ with transition operator K and centered transition operator $K' = K - J$, and $\pi_* = \min_{v \in V} \frac{\deg(v)}{2|E|}$. Given $\epsilon > \min(\pi_*^{-1/2}, \rho(K'))$, take $\ell = \left\lceil \frac{1}{4} \frac{\log(\epsilon^2 \pi_*)}{\log(\rho(\overline{K})/\epsilon)} \right\rceil$, and take $k = \lceil (\frac{1}{\epsilon})^{2\ell} \rceil$. Then for $R = k\ell + 1$, it holds that $2R$-local degree-2 SOS can deduce the bound $\rho(K') \le \frac{5}{2}\epsilon$. In particular, if we think of G as a* MAX-CUT *CSP, it holds that R-round Sherali–Adams can deduce the bound* $\mathrm{OBJ} \le \frac{1}{2} + \frac{5}{4}\epsilon$.

Again, Corollary 33 implies Theorem 1 since in simple graphs, $\log \frac{1}{\pi^*} = \Theta(\log n)$.

Proof of Theorem 32. The proof is a modification of the proof of Theorem 30. Letting S be the (k, ℓ)-spider vertices, instead of taking a signed stationary S-indexed random walk in G, we take two independent *unsigned* stationary S-indexed random walks, $\boldsymbol{\phi}_1$ and $\boldsymbol{\phi}_2$. For $j \in \{1, 2\}$, define $\mathbf{x}_j$ to be the S-indexed vector with ith coordinate equal to $\mathsf{X}_{\boldsymbol{\phi}_j(i)}$, and write $\dot{\mathbf{x}}$ for the concatenated vector $(\mathbf{x}_1, \mathbf{x}_2)$. Also, for $0 < \theta < 1$ a parameter[12] slightly less than 1, let Ψ be the PSD matrix from Corollary 28, and define the PSD block-matrix

$$\dot{\Psi} = \tfrac{1}{2}\left(\begin{array}{c|c} \frac{1}{\theta}\Psi & -\Psi \\ \hline -\Psi & \theta\Psi \end{array}\right).$$

Then as before, the $2R$-local, degree-2 SOS proof system can derive

$$0 \le \langle \dot{\Psi}, \mathbf{E}\, \dot{\mathbf{x}}\dot{\mathbf{x}}^\top \rangle = \iota \langle \Psi, \mathsf{Y} \rangle - \langle \Psi, \mathsf{Z} \rangle, \quad \text{where } \iota = \tfrac{1/\theta + \theta}{2}, \quad \mathsf{Y} = \mathbf{E}[\mathbf{x}\mathbf{x}^\top], \quad \mathsf{Z} = \mathbf{E}[\mathbf{x}_1\mathbf{x}_2^\top], \tag{15}$$

and $\mathbf{x}$ (which will play the role of $\overline{\mathbf{x}}$) denotes the common distribution of $\mathbf{x}_1$ and $\mathbf{x}_2$. Similar to Equations (11) and (12), we now have

$$\mathsf{Y} = \sum_{d=0}^{2\ell} \langle \mathsf{X}, K^d \mathsf{X} \rangle_\pi A^{(d)},$$

and by independence of $\mathbf{x}_1$ and $\mathbf{x}_2$ we have

$$\mathsf{Z} = \langle 1, X \rangle_\pi^2 \cdot J = \langle 1, X \rangle_\pi^2 \cdot \sum_{d=0}^{2\ell} A^{(d)}.$$

Thus applying Corollary 28 to Inequality (15), our proof system can derive

$$0 \le \iota \cdot \Big(c_0 \langle \mathsf{X}, \mathsf{X} \rangle_\pi + k^{1/(2\ell)} \langle \mathsf{X}, K\mathsf{X} \rangle_\pi + \tfrac{1}{2}(k-1) \langle \mathsf{X}, K^{2\ell}\mathsf{X} \rangle_\pi \Big) \\ - \mathop{\mathbf{E}}_{\boldsymbol{u} \sim \pi} [\mathsf{X}_{\boldsymbol{u}}]^2 \cdot \Big(c_0 + k^{1/(2\ell)} + \tfrac{1}{2}(k-1) \Big). \tag{16}$$

By selecting θ appropriately, we can arrange for the factor $c_0 + k^{1/(2\ell)} + \frac{1}{2}(k-1)$ on the right to equal $\iota \cdot \frac{1}{2}(k-1)$. Inserting this choice into Inequality (16) and then dividing through by ι, we conclude that the proof system can derive

$$0 \le c_0 \langle \mathsf{X}, \mathsf{X} \rangle_\pi + k^{1/(2\ell)} \langle \mathsf{X}, K\mathsf{X} \rangle_\pi + \tfrac{1}{2}(k-1) \big(\langle \mathsf{X}, K^{2\ell}\mathsf{X} \rangle_\pi - \langle 1, X \rangle_\pi^2 \big),$$

cf. Inequality (13). Recalling now that K has the constantly-1 function as an eigenvector, with eigenvalue 1, we have the identity

$$\langle \mathsf{X}, K^{2\ell}\mathsf{X} \rangle_\pi - \langle 1, X \rangle_\pi^2 = \langle \mathsf{X}, (K - J)^{2\ell}\mathsf{X} \rangle_\pi.$$

Now the remainder of the proof is just as in Theorem 27, with $K - J$ in place of $\overline{K}$, except we do not have the step of repeating the derivation with $-\overline{K}$ in place of $\overline{K}$. ◀

[12] This parameter is introduced to fix a small annoyance; the reader might like to imagine $\theta = 1$ at first.

3.2 A technical construction of coefficients on the spider

Proof of Theorem 27. We are considering the (k,ℓ)-spider graph on vertex set S. We write V_t for the set of all vertices at distance t from the root (so $|V_0|=1$ and $|V_t|=k$ for $1\le t\le \ell$). We will be considering vectors in $\mathbb{R}^S$, with coordinates indexed by the vertex set S. For $0\le t\le \ell$ define the vector

$$\mu_t = \operatorname*{avg}_{i\in V_t}\{\alpha^t e_i\},$$

where $e_i=(0,\dots,0,1,0,\dots,0)$ is the vector with the 1 in the ith position. Further define vectors

$$\begin{aligned}\chi &= \mu_0+\mu_1,\\ \psi_t &= \mu_t-\mu_{t+2} \text{ for } 0\le t<\ell,\end{aligned}$$

with the understanding that $\mu_{\ell+1}=0$. Next, define the PSD matrix

$$\widetilde{\Psi} = \chi\chi^\top + \sum_{t=0}^{\ell-1}\psi_t\psi_t^\top.$$

This will almost be our desired final matrix Ψ. Let us now compute

$$\langle\widetilde{\Psi}, A^{(d)}\rangle = \chi^\top A^{(d)}\chi + \sum_{t=0}^{\ell-1}\psi_t^\top A^{(d)}\psi_t.$$

To do this, we observe that

$$\mu_s^\top A^{(d)}\mu_t = \alpha^{s+t}\operatorname*{\mathbf{Pr}}_{\boldsymbol{i}\sim V_s,\ \boldsymbol{j}\sim V_t}[\operatorname{dist}_S(\boldsymbol{i},\boldsymbol{j})=d],$$

and

$$\mu_0^\top A^{(d)}\mu_t = \mu_t^\top A^{(d)}\mu_0 = \begin{cases}\alpha^t & \text{if } d=t,\\ 0 & \text{else;}\end{cases}$$

$$\text{and for } s,t>0,\quad \mu_s^\top A^{(d)}\mu_t = \begin{cases}(1/k)\alpha^{s+t} & \text{if } d=|s-t|,\\ (1-1/k)\alpha^{s+t} & \text{if } d=s+t,\\ 0 & \text{else.}\end{cases}$$

From this we can compute the following (with a bit of effort):

$$\begin{aligned}\langle\widetilde{\Psi}, A^{(0)}\rangle &= 2+(2/k)\alpha^2+(2/k)\alpha^4+\cdots+(2/k)\alpha^{2\ell-2}+(1/k)\alpha^{2\ell}\\ \langle\widetilde{\Psi}, A^{(1)}\rangle &= 2\alpha\\ \langle\widetilde{\Psi}, A^{(2)}\rangle &= -(2/k)\alpha^2-(2/k)\alpha^4-(2/k)\alpha^6-\cdots-(2/k)\alpha^{2\ell-2}\\ \langle\widetilde{\Psi}, A^{(2t+1)}\rangle &= 0,\quad 1\le t<\ell\\ \langle\widetilde{\Psi}, A^{(2t)}\rangle &= 0,\quad 1<t<\ell\\ \langle\widetilde{\Psi}, A^{(2\ell)}\rangle &= (1-1/k)\alpha^{2\ell}\end{aligned}$$

Now, for a parameter $\eta>0$ to be chosen shortly, we finally define the PSD matrix

$$\Psi = \frac{1}{2}\widetilde{\Psi}+\eta\mu_1\mu_1^\top.$$

We have

$$\langle \eta \mu_1 \mu_1^\top, A^{(d)} \rangle = \begin{cases} \eta(1/k)\alpha^2 & \text{if } d = 0, \\ \eta(1-1/k)\alpha^2 & \text{if } d = 2. \end{cases}$$

Therefore by carefully choosing

$$\eta = \frac{1}{k-1}\left(\frac{\alpha^{2\ell-2}-1}{\alpha^2-1}\right),$$

we get all of the desired inner products in the theorem statement. ◀

4 Lower Bounds

In this section, we show that degree-R Sherali–Adams cannot refute a random 2-XOR or MAX-CUT instance better than $\frac{1}{2} + \Omega(\frac{1}{R})$. This is a straightforward application of the framework of Charikar, Makarychev and Makarychev [15]. In that work, the authors show that if every subset of r points in a metric can be locally embedded into the unit sphere, then Goemans-Williamson rounding can be used to give a $\Theta(r)$-round Sherali–Adams feasible point. The upshot is the following theorem appearing in [15] (where it is stated in slightly more generality, for the 0/1 version of the cut polytope):

▶ **Theorem 34** (Theorem 3.1 in [15]). *Let (X, ρ) be a metric space, and assume that every $r = 2R+3$ points of (X, ρ) isometrically embed in the Euclidean sphere of radius 1. Then the following point is feasible for R-rounds of the Sherali–Adams relaxation for the cut polytope:*

$$\widetilde{\boldsymbol{E}}[x_i x_j] = 1 - \frac{2}{\pi}\arccos\left(1 - \frac{1}{2}\rho(i,j)^2\right).$$

▶ **Proposition 35.** *In any 2-XOR or MAX-CUT instance, R-rounds of Sherali–Adams cannot certify that*

$$\mathrm{OBJ}(x) < \frac{1}{2} + \frac{1}{\pi R} - \frac{1}{2R^2}$$

Proof. Suppose that we are given a 2-XOR (equivalently, MAX-CUT) instance on the graph G, so that on each edge $(i,j) \in E(G)$ we have the constraint $x_i x_j b_{ij} = 1$ for some $b_{ij} \in \{\pm 1\}$. Define the metric space on (X, ρ) as follows: let $X = \{x_1, \ldots, x_n\}$ have a point for each vertex of G, and set $\rho(x_i, x_j) = \sqrt{2\left(1 - b_{ij}\frac{1}{R}\right)}$.

We claim that any $r = 2R+3$ points of X embed isometrically into the Euclidean sphere of radius 1. To see this, fix a set $S \subset X$, and define the $|S| \times |S|$ matrix B_S so that

$$(B_S)_{ij} = \begin{cases} \frac{b_{ij}}{r} & \text{if } (i,j) \in E(G), \\ 0 & \text{otherwise.} \end{cases}$$

So long as $|S| \leq r$, the matrix $M_S = \mathbf{1} + B_S$ is diagonally dominant, and therefore positive semidefinite, so from the Cholesky decomposition of M_S we assign to each $x_i \in S$ a vector v_i so that $\|v_i\|_2 = 1$, and so that for every pair $x_i, x_j \in S$, $\|v_i - v_j\|^2 = 2 - 2b_{ij}\frac{1}{r} = \rho(i,j)^2$.

Applying Theorem 34, we have that the solution

$$\widetilde{\mathbf{E}}[x_i x_j] = 1 - \frac{2}{\pi}\arccos\left(1 - \frac{1}{2}\cdot 2\left(1 - b_{ij}\frac{1}{r}\right)\right) = 1 - \frac{2}{\pi}\arccos\left(b_{ij}\frac{1}{r}\right)$$

is feasible. For convenience, let $f(z) = 1 - \frac{2}{\pi}\arccos(z)$. We use the following properties of f:

▷ Claim 36. The function $f(z) = 1 - \frac{2}{\pi}\arccos(z)$ exhibits the rotational symmetry $f(z) = -f(-z)$, and further $f(z) \geq \frac{2}{\pi}z$ for $z \in [0,1]$.

We give the proof of the claim (using straightforward calculus) below. Now, because $f(z) = -f(-z)$, we have that

$$b_{ij} \cdot \widetilde{\mathbf{E}}[x_i x_j] = b_{ij} \cdot f\left(b_{ij}\frac{1}{r}\right) = f\left(\frac{1}{r}\right),$$

and using that for $z \in [0,1]$, $f(z) \geq \frac{2}{\pi}z \geq 0$,

$$\geq \frac{2}{\pi} \cdot \frac{1}{r}.$$

We conclude that $R = \frac{1}{2}(r-3)$ rounds of Sherali–Adams are unable to certify that $\mathrm{OBJ} < \frac{1}{2} + \frac{2}{\pi}\frac{1}{2R+3}$, as desired. ◀

Proof of Claim 36. The rotational symmetry follows from simple manipulations:

$$f(z) - (-f(-z)) = 2 - \frac{2}{\pi}(\arccos(z) + \arccos(-z)) = 2 - \frac{2}{\pi}\arccos(-1) = 0.$$

For the second claim, we use that the derivative of $f(z) - \frac{2}{\pi}z$ is positive in the interval $[0, \frac{1}{2}]$:

$$\frac{\partial}{\partial z}f(z) - \frac{2}{\pi}z = \frac{2}{\pi}\frac{1}{\sqrt{1-z^2}} - \frac{1}{2} > 0 \text{ for } |z| < 1,$$

and that at $z = 0$, $f(z) - \frac{2}{\pi}z = 0$. ◁

5 Refutation for any Boolean CSP

In this section, we argue that R-round Sherali–Adams can also refute any non-trivial Boolean CSP. First, for any predicate $P : \{\pm 1\}^k \to \{0,1\}$ we define a parameterized distribution over the CSP with constraints from P:

▶ **Definition 37.** *Let $P : \{\pm 1\}^k \to \{0,1\}$ be a predicate. Then we define a* random instance of P on n vertices with m expected clauses *to be an instance sampled as follows: define $p = \frac{m}{n^k}$, and for each ordered multiset $S \subset [n]$ with $|S| = k$, independently with probability p we sample a uniformly random string $\zeta_S \in \{\pm 1\}^k$ and add the constraint that $P(x^S \odot \zeta_S) = 1$, where $\odot$ denotes the entry-wise (or Hadamard) product.*

This is one of several popular models, and in our case it is the most convenient to work with. By employing some manipulations, results from this model transfer readily to the others (see for example Appendix D of [2] for details).

Our result is as follows:

▶ **Theorem 38.** *Suppose that $P : \{\pm 1\}^k \to \{0,1\}$ and that $\delta, \epsilon > 0$ are fixed constants. Let $\boldsymbol{E}[P]$ be the probability that a random $x \in \{\pm 1\}^k$ satisfies P. Then with high probability, for a random instance $\mathcal{I}$ of P on n variables with $m \geq n^{\lceil k/2 \rceil + \delta}$ expected clauses, the R-round Sherali–Adams proof system can certify that $\mathrm{OBJ}_{\mathcal{I}}(x) \leq \boldsymbol{E}[P] + \epsilon$ when $R = O_{\epsilon,\delta,k}(1)$ rounds. More specifically, $R = k\ell\left(\frac{3 \cdot 2^{k/2-1}}{\epsilon}\right)^{2\ell} + k$ for $\ell = \lceil \lceil \frac{k}{2} \rceil \frac{1}{2\delta} \rceil$.*

We can also prove a more fine-grained result, to obtain strong refutation at lower clause densities when the predicate has certain properties.

▶ **Definition 39.** *We say that a predicate $P : \{\pm 1\}^k \to \{0,1\}$ is η-far from t-wise supporting if every t-wise uniform distribution has probability mass at least η on the set of unsatisfying assignments $P^{-1}(0)$.*

▶ **Theorem 40.** *Suppose that $P : \{\pm 1\}^k \to \{0,1\}$ is η-far from t-wise supporting, and that $\delta, \epsilon > 0$. Then with high probability, for a random instance $\mathcal{I}$ of P on n variables and $m \geq n^{\lceil t/2 \rceil + \delta}$ expected clauses, the R-round Sherali–Adams proof system can certify that $\mathrm{OBJ}_{\mathcal{I}}(x) \leq 1 - \eta + \epsilon$ with $R = O_{\epsilon,\delta,t}(1)$ rounds. More specifically, $R = t\ell \left(\frac{3 \cdot 2^{t/2-1}}{\epsilon}\right)^{2\ell} + t$ for $\ell = \lceil \lceil \frac{t}{2} \rceil \frac{1}{2\delta} \rceil$.*

Following the strategy introduced in [2], we will do this by first refuting weighted random instances of k-XOR for $k \geq 1$. After this, any predicate $P : \{\pm 1\}^k \to \{0,1\}$ can be decomposed according to its Fourier decomposition, which will yield a weighted sum of t-XOR instances for $t \leq k$, and our proof system will refute each individually.

5.1 Higher-arity XOR

Ultimately, we will reduce each k-CSP to a sum over weighted t-XOR instances with $t \leq k$:

▶ **Definition 41.** *Let $\mathcal{W}$ be a distribution over signed integers. We say that $\mathcal{I}$ is a* random k-XOR instance weighted according to $\mathcal{W}$ *if it is sampled as follows: for each ordered multiset $S \subset [n]$ with $|S| = k$, we take a b_S to be equal to a uniformly random sample from $\mathcal{W}$, and finally set the objective function to be $\sum_S b_S \cdot x^S$.*

Following the standard strategy introduced by [30, 26] and subsequently honed in many works, we will reduce refuting these t-XOR instances to refuting 2-XOR instances.

5.1.1 Even k-XOR

In this case, we perform a standard transformation to view the k-XOR instance as a 2-XOR instance on super-vertices given by subsets of vertices of size $k/2$.

▶ **Definition 42.** *Suppose $k > 1$ is an integer and $\mathcal{I}$ is a $2k$-XOR instance on n variables $x_1, \ldots, x_n$, with objective $\sum_{U \in [n]^{2k}} b_U \cdot x^U$ where the sum is over ordered multisets $U \subset [n], |U| = 2k$. Then we let its* flattening, *$\mathcal{I}_{flat}$, be the 2-XOR instance on n^k variables given by associating a new variable y_S for each ordered multiset $S \subset [n], |S| = k$, and for each $U \subset [n]$ with $|U| = 2k$, choosing the partition of U into the ordered multisets S, T with S containing the first k elements and T containing the last k, taking the objective function $\sum_{S,T} b_U \cdot y_S y_T$.*

▶ **Lemma 43.** *Suppose that $\mathcal{I}$ is a $2k$-XOR instance, and let $\mathcal{I}_{flat}$ be the 2-XOR instance given by its flattening. Then if the R-round Sherali–Adams proof system can certify that $\mathrm{OBJ}_{\mathcal{I}_{flat}}(x) \leq c$, then the $k \cdot R$-round Sherali–Adams proof system can certify that $\mathrm{OBJ}_{\mathcal{I}}(x) \leq c$.*

Proof. Every degree-R Sherali–Adams proof for $\mathcal{I}_{flat}$ can be transformed into a Sherali–Adams proof of degree at most kR for $\mathcal{I}$ by applying the transformation $y_S = \prod_{i \in S} x_i = x^S$. Further, this transformation exactly relates the objective functions of $\mathcal{I}_{flat}$ and $\mathcal{I}$. This proves the claim. ◀

If the $2k$-XOR instances that we start with are random weighted instances, then their flattenings are also random weighted 2-XOR instances.

▷ Claim 44. Suppose that $\mathcal{I}$ is a random $2k$-XOR instance on n vertices weighted according to $\mathcal{W}$. Then the flattening $\mathcal{I}_{flat}$ is a random 2-XOR instance on n^k vertices weighted according to $\mathcal{W}$.

Proof. This fact is immediate, since the ordered multisets $U \subset [n]$, $|U| = 2k$ are in bijection with ordered pairs of multisets $S, T \subset [n]$, $|S| = |T| = k$. ◁

We will require the following proposition, which applies our main theorem in the context of random k-XOR instances with random weights from well-behaved distributions.

▶ **Proposition 45.** *Suppose that $\mathcal{W}$ is a distribution over integers which is symmetric about the origin, and let $n, k \geq 1$ be positive integers. Let $\boldsymbol{E}$ denote the expectation under the measure $\mathcal{W}$, and let $\sigma^2 \geq Ew^2$ be a bound on the variance. Furthermore, suppose that*

- *The expected absolute value is at least $\boldsymbol{E}|w| \gg \sigma\sqrt{\frac{\log n}{n^k}}$,*
- *With high probability over n^{2k} i.i.d. samples $w_1, \ldots, w_{n^{2k}} \sim \mathcal{W}$, $\max_{i \in [n^{2k}]} |w_i| \leq M \ll \sigma^2 n^k$.*

Now, define

$$\rho = O\left(\frac{\sigma \log N}{\boldsymbol{E}|w|\sqrt{n^k}} \cdot \max(1, \tfrac{M}{\sqrt{n^k}})\right).$$

Then if $\mathcal{I}$ is a random $2k$-XOR instance on n variables weighted according to $\mathcal{W}$, with high probability $\mathcal{I}$ has $\boldsymbol{E}|w| \cdot n^{2k} \pm O(\sigma n^k \sqrt{\log n})$ constraints. Further, choosing $\ell \in \mathbb{N}_+$ large enough so that $n^{k/4\ell}\rho \leq \frac{1}{2}\epsilon^{2\ell}$ and setting $R = 2k \cdot \ell \cdot \left(\frac{1}{\epsilon}\right)^{2\ell}$, R rounds of Sherali–Adams can deduce the bound $\mathrm{OBJ}_{\mathcal{I}}(x) \leq \frac{1}{2} + \frac{3}{2}\epsilon$.

To prove the above, we require the following standard matrix Bernstein inequality:

▶ **Theorem 46** (Theorem 6.6.1 in [48]). *Let $A_1, \ldots, A_m \in \mathbb{R}^{N \times N}$ be independent random matrices, with $\boldsymbol{E} A_i = 0$ for all $i \in [m]$ and $\|A_i\| \leq M$ for all $i \in [m]$. Let $A = \sum_{i \in [m]} A_i$ denote their sum, and suppose that $\|\boldsymbol{E} AA^\top\|, \|EA^\top A\| \leq \sigma^2$. Then*

$$\boldsymbol{Pr}\left(\|A\| \geq t\right) \leq N \cdot \exp\left(\frac{1}{2}\frac{-t^2}{\sigma^2 + \frac{1}{3}Mt}\right).$$

Proof of Proposition 45. Given a weighted $2k$-XOR instance on n variables with weights from $\mathcal{W}$, we consider its flattening $\mathcal{I}_{flat}$ with objective function $\mathrm{OBJ}(x) = \frac{1}{m}\sum_{i,j \in [N]} \frac{1}{2}(1 + b_{ij}x_i x_j)$ for m the absolute sum of weights, we construct its signed adjacency matrix as follows: first take the matrix W defined so that $W_{i,j} = b_{ij}$, and obtain a new matrix $B = \frac{1}{2}(W + W^\top)$. For any x, applying Lemma 43 we have that $\frac{1}{2} + \frac{1}{2m}x^\top Bx = \mathrm{OBJ}_{\mathcal{I}}(x)$.

Since $\mathcal{W}$ is a distribution over integers, $2B$ has signed integer entries. We think of $2B$ as defining a multigraph G on n^k vertices with signed edges, so that there are $2 \cdot |B_{ij}|$ multiedges between $i, j \in [n^k]$, each with sign $\mathrm{sgn}(B_{ij})$. Let $2 \cdot D$ be the degree matrix of G, let $A = |2B|$ be the adjacency matrix of G, let $\Xi = \mathrm{sgn}(B)$ be the matrix of signs of B, and let $K = (2D^{-1})A = D^{-1}B \otimes \Xi$ be the transition matrix for the random walk on G.

To apply Corollary 31, we must upper bound the spectral radius of $\Xi \circ K = D^{-1}B$, as well as bound the minimum degree of G and the total number of edges. We will use the bound

$$\|D^{-1}B\|_{op} \leq \|D^{-1}\|_{op} \cdot \|B\|_{op} \leq \frac{1}{\pi^*}\|B\|_{op}.$$

First, we bound $\|B\|_{op}$. Take B' to be the truncated version of B, so that $B'_{i,j} = \mathrm{sgn}(B_{i,j}) \cdot \max(|B_{i,j}|, M)$. Thinking of the matrix B' as the sum of $\binom{n^k}{2} + n^k$ symmetric matrices, one for each pair $i, j \in [n^k]$, satisfies the requirements of Theorem 46. We have that $\mathbf{E}[B'B'^{\top}] \preceq n^k\sigma^2 \cdot \mathbf{1}$, so applying Theorem 46 with $t = O\max(\sqrt{\sigma^2 n^k \log n}, M \log n)$ we get that with high probability,

$$\|B\|_{op} \leq O\left(\max\left(\sqrt{\sigma^2 n^k \log n}, M \log n\right)\right),$$

where we have also used that with high probability $B = B'$ by the properties of $\mathcal{W}$.

Now, we bound the sum of degrees $2m$ and the minimum degree $d_{\min}$. We have that the total sum of the degrees is given by $2m = \sum_{i,j\in[n^k]} |b_{ij}|$ with $b_{ij} \sim \mathcal{W}$. By a Bernstein inequality,

$$\mathbf{Pr}\left(\left|2m - n^{2k}\,\mathbf{E}\,|w|\right| \geq s\right) \leq 2\exp\left(-\frac{1}{2}\frac{s^2}{n^{2k}\cdot\mathbf{E}\,w^2 + \frac{1}{3}Ms}\right),$$

so since by assumption $\sigma^2 n^{2k} \gg M$, setting $s = O(\sigma n^k\sqrt{\log n})$ we have that with high probability

$$2m = n^{2k}\,\mathbf{E}\,|w| \pm O(\sigma n^k\sqrt{\log n}). \tag{17}$$

By our assumptions on $\mathcal{W}$ we have that for every $i \in [n^k]$, $\mathbf{E}\deg_G(i) = n^k\,\mathbf{E}\,|w|$. Applying a Bernstein inequality gives that

$$\mathbf{Pr}[\deg_G(i) \leq n^k\,\mathbf{E}\,|w| - t] \leq \exp\left(-\frac{1}{2}\cdot\frac{t^2}{n^k\sigma^2 + \frac{1}{3}Mt}\right),$$

so using that $M \leq n^k\sigma^2$ and taking $t = O\left(\sqrt{n^k\sigma^2\log n}\right)$ we get that $d_{\min} = n^k\cdot\mathbf{E}\,|w| \pm O(\sqrt{n^k\sigma^2\log n})$ with high probability. This gives that with high probability,

$$\pi^* = \frac{d_{\min}}{2m} \geq \frac{n^k\cdot\mathbf{E}\,|w| - O(\sigma\sqrt{n^k\log n})}{n^{2k}\cdot\mathbf{E}\,|w| + O(\sigma n^k\sqrt{\log n})} \geq \frac{1}{n^k}\cdot(1-o(1))$$

$$\|\Xi\circ K\|_{op} \leq \frac{\|B\|_{op}}{d_{\min}} \leq \frac{O(\sigma\sqrt{n^k}\log n)\cdot\max(1,\frac{M}{\sqrt{n^k}})}{n^k\cdot\mathbf{E}\,|w| - O(\sigma\sqrt{n^k\log n})} \leq O\left(\frac{\sigma\log n}{\mathbf{E}\,|w|\sqrt{n^k}}\right)\cdot\max(1,\tfrac{M}{\sqrt{n^k}})$$

where we have used that $\sigma\sqrt{\log n} \ll \sqrt{n^k}\,\mathbf{E}\,|w|$.

Now, the result follows by applying Corollary 31 and Lemma 43. ◀

5.1.2 Odd k-XOR

For odd integers k, k-XOR instances do not have the same natural, symmetric flattenings. Instead, we define what we call a *lift*:

▶ **Definition 47.** *Suppose $k \geq 1$ is an integer and $\mathcal{I}$ is a $(2k+1)$-XOR instance on n variables $x_1, \ldots, x_n$, with objective $\sum_{U\in[n]^{2k+1}} b_U \cdot x^U$. Then we let its* lift*, $\mathcal{I}_{lift}$, be the bipartite 2-XOR instance on parts each containing n^{k+1} variables created as follows:*

- *Create new variables $w_1, \ldots, w_n$*
- *For each $U \in [n]^{2k+1}$, choose a random index $i_U \in [n]$ and add modify the objective to $\sum_{U\in[n]^{2k+1}} b_U \cdot x^U \cdot w_{i_U}$*

- *For each ordered multiset S associate a new variable y_S, and for each ordered multiset $T \in [n]^k$ and index $i \in [n]$ associate a new variable $z_{T,i}$ We understand $y_S = \prod_{i \in S} x_i$, and $z_{T,i} = \left(\prod_{j \in T} x_j\right) \cdot w_i$.*
- *For each $U \in [n]^{2k+1}$, we take the ordered multiset $V = (U, i_U)$ and assign it a new coefficient $b'_V = b_U$. For the remaining b'_V, we set $b'_V = 0$.*

Finally, $\mathcal{I}_{lift}$ is the instance with the objective function $\sum_{S \in [n]^{k+1}, T \in [n]^k, i \in [n]} b'_{S \cup T \cup i} \cdot y_S z_{T,i}$.

We obtain a statement analogous to Lemma 43 for odd k-XOR:

▶ **Lemma 48.** *Suppose that $\mathcal{I}$ is a weighted $(2k+1)$-XOR instance, and let $\mathcal{I}_{flat}$ be the bipartite 2-XOR instance given by its flattening. Then if the R-round Sherali–Adams proof system can certify that $\mathrm{OBJ}_{\mathcal{I}_{flat}}(x) \leq c$, then the $(k+1) \cdot R$-round Sherali–Adams proof system can certify that $\mathrm{OBJ}_{\mathcal{I}}(x) \leq c$.*

Proof. The only modification to the proof of Lemma 48 is that for $z_{T,i}$ we substitute $z_{T,i} = x^T$ (where we have implicitly substituted $w_i = 1$ for all $i \in [n]$). ◀

However, the lifting procedure does not preserve the weighting distribution $\mathcal{W}$, because of the step in which a random index i_U is chosen to lift U. For this reason, we prove an analog of Proposition 45:

▶ **Proposition 49.** *Suppose that $\mathcal{W}$ is a distribution over integers which is symmetric about the origin, and let $n, k \geq 1$ be integers. Let $\boldsymbol{E}$ denote the expectation under the measure $\mathcal{W}$, and let $\sigma^2 \geq Ew^2$ be a bound on the variance. Furthermore, suppose that*

- *The expected absolute value is at least $\boldsymbol{E}\,|w| \gg \sigma\sqrt{\frac{\log n}{n^k}}$,*
- *With high probability over n^{2k+1} i.i.d. samples $w_1, \ldots, w_{n^{2k+1}} \sim \mathcal{W}$, $\max_{i \in [n^{2k+1}]} |w_i| \leq M \ll \sigma^2 n^k$.*

Now, define

$$\rho = O\left(\frac{\sigma \log n}{\boldsymbol{E}\,|w| \sqrt{n^k}} \cdot \max(1, \tfrac{M}{\sqrt{n^k}})\right).$$

Then if $\mathcal{I}$ is a random $(2k+1)$-XOR instance on n variables weighted according to $\mathcal{W}$, with high probability it has $\boldsymbol{E}\,|w| \cdot n^{2k+1} \pm O(\sigma n^k \sqrt{\log n})$ constraints. Furthermore, choosing $\ell \in \mathbb{N}_+$ large enough so that $n^{(k+1)/4\ell}\rho \leq \frac{1}{2}\epsilon^{2\ell}$ and setting $R = (2k+2)\ell \cdot \left(\frac{1}{\epsilon}\right)^{2\ell}$, R rounds of Sherali–Adams can deduce the bound $\mathrm{OBJ}_{\mathcal{I}}(x) \leq \frac{1}{2} + \frac{3}{2}\epsilon$.

Proof. The thread of the proof is the same as that of Proposition 45. We will refute $\mathcal{I}_{lift}$, since by Lemma 48 this is sufficient. We begin by associating with $\mathcal{I}_{lift}$ a multigraph G (which we may do because $\mathcal{W}$ is a distribution over integers). The multigraph G is a bipartite graph, with one bipartition corresponding to variables y_S for $S \in [n]^{k+1}$, and one bipartition corresponding to variables $z_{T,i}$ for $T \in [n]^k$ and $i \in [n]$. We let the block matrix B be (the $\frac{1}{2}$-scaled) signed adjacency matrix of G, let Ξ be the matrix of signs, D be the diagonal degree matrix, and $K \circ \Xi = D^{-1}B$ be the signed transition matrix of the random walk on G. In order to apply Corollary 31 we must bound $\|K \circ \Xi\|$ and $\pi^* = d_{\min}(G)/2m$.

First, we bound the vertex degrees. For a vertex of the form (T, i), the expected value of the incident edge $(S, T \cup i)$ is $b_{S,T} \cdot \frac{1}{n}$, where $b_{S,T}$. The degree of $T \cup i$ is simply the sum

$$\deg_G(T \cup i) = \sum_{S \in [n]^{k+1}} |b_{S,T \cup i}|,$$

a sum of independent random variables with expectation $\frac{1}{n}\mathbf{E}|w|$ and variances $\frac{1}{n}\sigma^2$. Applying a Bernstein inequality, we have that

$$\mathbf{Pr}\left(\left|\deg_G(T\cup i) - \frac{1}{n}\cdot n^{k+1}\,\mathbf{E}[|w|]\right| \geq t\right) \leq 2\exp\left(\frac{1}{2}\frac{-t^2}{n^k\sigma^2 + \frac{1}{3}Mt}\right),$$

so taking $t = O(\sqrt{\sigma^2 n^k \log n})$ (and using that $M \ll n^k\sigma^2$), we have that the degree of $T\cup i$ vertices is $\deg_G(T\cup i) = n^k\,\mathbf{E}|w| \pm O(\sqrt{\sigma^2 n^k \log n})$ with high probability.

A similar argument applies to the S vertices; the total degree of such a vertex is

$$\deg_G(S) = \sum_{T\in[n]^k} |\sum_{i\in[n]} b_{S,T\cup i}|,$$

since only one of the $b_{S,T\cup i}$ will be nonzero. The inner sums are independent random variables with mean $\mathbf{E}|w|$ and variance σ^2, therefore

$$\mathbf{Pr}\left(\left|\deg_G(S) - n^k\,\mathbf{E}|w|\right| \geq t\right) \leq 2\exp\left(\frac{1}{2}\frac{-t^2}{n^k\sigma^2 + \frac{1}{3}Mt}\right),$$

so taking $t = O(\sqrt{\sigma^2 n^k \log n})$ (and using that $M \ll n^k\sigma^2$), we have that the degree of S vertices is also $\deg_G(S) = n^k\,\mathbf{E}|w| \pm O(\sqrt{\sigma^2 n^k \log n})$ with high probability.

We finally bound $\|K\circ\Xi\| \leq \|D^{-1}\|\cdot\|B\|$. As before, B is a sum of independent symmetric matrices, one for each coefficient b_U from $\mathcal{I}$. That is, we can define matrices B_U for each $U\in[n]^{2k+1}$ with $U = S,T$ for $S\in[n]^{k+1}, T\in[n]^k$ where B_U has a number $b_U\sim\mathcal{W}$ in one off-diagonal block entry $(S, T\cup i)$ and the other off-diagonal block entry $(T\cup i, S)$ for a randomly chosen $i\in[n]$. Thus, $\mathbf{E}\,B_U B_U^\top$ is a diagonal matrix with $\frac{1}{n}\sigma^2$ on each diagonal of the form $(T\cup i, T\cup i)$ and σ^2 on each block diagonal of the form (S,S). We then have that $\mathbf{E}\,BB^\top \preceq n^k\sigma^2\mathbf{1}$, since for each S there is a sum over n^k matrices B_U and for each $T\cup i$ there is a sum over n^{k+1} matrices B_U. Applying Theorem 46 by using the same truncation trick again, we have that

$$\|B\| \leq O\left(\max\left(\sqrt{\sigma^2 n^k \log n}, M\log n\right)\right),$$

and from this we have that with high probability,

$$\|K\circ\Xi\| \leq \frac{1}{\deg_{\min}(G)} \leq O\left(\frac{\sigma\log n}{\mathbf{E}|w|\sqrt{n^k}}\right)\cdot\max(1, \tfrac{M}{\sqrt{n^k}}), \tag{18}$$

$$\pi^* = \frac{\deg_{\min}(G)}{2m} = \frac{1}{n^{k+1}}\cdot(1\pm o(1)) \tag{19}$$

After which we can apply Corollary 31. ◀

5.2 From Boolean CSPs to k-XOR

Following [2], we prove Theorem 38 via reduction to XOR.

Proof of Theorem 38. Given a random instance of the CSP defined by the predicate P, and $p = n^{-\lfloor k/2\rfloor+\delta}$, a Bernstein inequality gives us that the number of constraints m is with high probability given by $m = n^{\lceil k/2\rceil+\delta} \pm 10\sqrt{n^{\lceil k/2\rceil+\delta}\log n}$. Set $\ell = \lceil\frac{1}{2\delta}\rceil$.

Since P is a Boolean predicate, we can write P in its Fourier expansion:

$$P(z) = \sum_{\alpha\subseteq[k]} \widehat{P}(\alpha)\prod_{i\in\alpha} z_i.$$

Using this expansion, we re-write the objective function. Recall that $[n]^k$ is the set of all ordered multisets of k elements of $[n]$. For each $S \in [n]^k$, let b_S be the 0/1 indicator that there is a constraint on S. Then, if the total number of constraints is m,

$$\begin{aligned}\mathrm{OBJ}_{\mathcal{I}}(x) &= \frac{1}{m}\sum_{S\in[n]^k} b_S \cdot P(x^S \odot \zeta_S) \\ &= \frac{1}{m}\sum_{S=\{i_1,\ldots,i_k\}\in[n]^k}\sum_{\alpha\subseteq[k]} b_S \cdot \widehat{P}(\alpha)\prod_{a\in\alpha} x_{i_a}(\zeta_S)_{i_a} \\ &= \frac{1}{m}\sum_{\alpha\subseteq[k]}\widehat{P}(\alpha)\cdot\sum_{T\in[n]^{|\alpha|}}\left(\sum_{S\in[n]^k, S|_\alpha=T} b_S\cdot\prod_{a\in\alpha}(\zeta_S)_{i_a}\right)\cdot x^T. \end{aligned} \tag{20}$$

Now, define for each $\alpha \subseteq [k]$ with $|\alpha| = t > 0$ the t-XOR instance

$$\mathcal{I}^\alpha(x) = \frac{1}{m}\sum_{T\in[n]^t}\left(\sum_{S\in[n]^k,S|_\alpha=T} b_S\cdot\prod_{a\in\alpha}(\zeta_S)_{i_a}\right)\cdot x^T = \frac{1}{m}\sum_{T\in[n]^t} w_T\cdot x^T,$$

where we have taken $w_T = \sum_{S\in[n]^k,S|_\alpha=T} b_S\cdot\prod_{a\in\alpha}(\zeta_S)_{i_a}$. So that from Equation (20),

$$\mathrm{OBJ}_{\mathcal{I}}(x) = \sum_{\alpha\subseteq[k]}\widehat{P}(\alpha)\cdot\mathcal{I}^\alpha(x). \tag{21}$$

Let $\mathcal{W}_{n^{k-t}}$ be the distribution defined so that $w \sim \mathcal{W}_t$ is a sum of n^{k-t} independent variables taking value $\{\pm 1\}$ with probability p and value 0 otherwise. Since for each $S \supseteq T$, the quantity $\prod_{a\in\alpha}(\zeta_S)_{i_a}$ is an independent uniform sign in $\{\pm1\}$ and b_S is an independent Bernoulli-p variable, we have that the coefficients w_T in $\mathcal{I}^\alpha$ are i.i.d. from $\mathcal{W}_{n^{k-t}}$. The following lemma establishes some properties of $\mathcal{W}_N$ (we will prove the lemma in Appendix A):

▶ **Lemma 50.** *Let $\mathcal{W}_N(p)$ be the distribution defined so that $X \sim \mathcal{W}_N$ is given by $X = \sum_{t=1}^N Y_t\cdot Z_t$, where the $\{Y_t\}_t, \{Z_t\}_t$ are i.i.d with $Y_t \sim \mathrm{Ber}(p)$ and $Z_t \sim \{\pm1\}$. Then for $X \sim \mathcal{W}_N(p)$, $\boldsymbol{E}\,X = 0$ and $\boldsymbol{E}\,X^2 = pN$. Further, so long as $pN \geq 1$, $\boldsymbol{E}\,|X| \geq \frac{2}{e^{3/2}}\sqrt{pN}$, and $\boldsymbol{Pr}(|X| > 2t\sqrt{pN}) \leq 2\exp\left(-t^2\right)$. Otherwise, if $pN \leq 1$, $\boldsymbol{E}\,|X| \geq \frac{1}{2e}\log\frac{1}{1-pN}$, and $\boldsymbol{Pr}(|X| \geq 1+t) \leq \exp\left(-\frac{1}{2}t\right)$.*

From Lemma 50, we have that $\mathbf{E}\,w_T^2 = pn^{k-t}$, and by Cauchy-Schwarz $\mathbf{E}\,|w_T| \leq \sqrt{\mathbf{E}\,w_T^2}$. Let m_α be the total absolute weight of constraints in $\mathcal{I}^\alpha$,

$$m_\alpha = \sum_T |w_T|.$$

Notice that in all cases, $m_\alpha \leq m$.

Now, we show that SA can certify upper bounds on $|\mathcal{I}^\alpha(x)|$ for every α. First, consider α with $|\alpha| = t = 1$. In this case, Sherali–Adams with $R = 1$ can certify that

$$\mathcal{I}^\alpha(x) = \frac{1}{m}\sum_{i\in[n]} w_i\cdot x_i \leq \frac{1}{m}\sum_{i\in[n]}|w_i| = \frac{m_\alpha}{m},$$

From an application of Bernstein's inequality (the same as in the proof of Proposition 45), $m_\alpha \leq n\cdot\sqrt{\mathbf{E}\,w_T^2} + \sqrt{pn^k\log n}$ with high probability whenever $pn^k \gg \sqrt{pn^{k-1}}$, and applying our bound on m we conclude that with high probability SA will certify that

$$\mathcal{I}^\alpha(x) \leq \frac{n\cdot\sqrt{pn^{k-1}}(1+o(1))}{pn^k(1\pm o(1))} \leq \frac{2}{\sqrt{pn^{k-1}}}$$

The lower bound on $\mathcal{I}^\alpha(x)$ follows from identical reasoning with its negation, so we can conclude that with high probability SA can certify that

$$|\mathcal{I}^\alpha(x)| \le \frac{2}{\sqrt{pn^{k-1}}} = o(1).$$

Now, we tackle α with $|\alpha| = t$ for $2 \le t \le k$. We will verify that the conditions of Propositions 45 and 49 hold. First, consider the α with $|\alpha| = t$ for $pn^{k-t} \ge 1$. From Lemma 50, in this case we have that $\mathbf{E}\,|w_T| \ge \frac{2}{e^{3/2}}\sqrt{pn^{k-t}}$, and with high probability, $|w_T| \le O(\sqrt{tpn^{k-t}\log n})$ for all $T \in [n]^t$. Letting $M = O(\sqrt{tpn^{k-t}\log n})$, and $\sigma^2 = pn^{k-t}$, we meet the conditions for Proposition 45:

$$M \le O(\sqrt{pn^{k-t}}) \ll pn^{k-t}\cdot n^{\lfloor t/2\rfloor} = \sigma^2 n^{\lfloor t/2\rfloor}$$
$$\mathbf{E}\,|w_T| \ge \frac{2}{e^{3/2}}\sqrt{pn^{k-t}} \gg \sqrt{\frac{pn^{k-t}\log n}{n^{\lfloor t/2\rfloor}}} = \sqrt{\frac{\sigma^2\log n}{n^{\lfloor t/2\rfloor}}}$$

so long as $pn^{k-t} \ge 1$, which we have assumed. So applying Propositions 45 and 49 to both $\frac{m}{m_\alpha}\mathcal{I}^\alpha$ and $-\frac{m}{m_\alpha}\mathcal{I}^\alpha$, we have

$$\rho = O\left(\frac{\sigma\log n}{\mathbf{E}\,|w_T|\sqrt{n^{\lfloor t/2\rfloor}}}\right)\cdot\max(1, \tfrac{M}{\sqrt{n^{\lfloor t/2\rfloor}}}) \le O\left(\frac{\log n}{\sqrt{n^{\lfloor t/2\rfloor}}}\right)\cdot\max\left(1, \frac{\sqrt{pn^{k-t}}}{n^{\lfloor t/2\rfloor}}\right)$$

and so long as $\frac{m_\alpha}{m}\cdot n^{\lceil t/2\rceil/4\ell}\rho \le \frac{1}{2}\epsilon^{2\ell}$, with high probability over $\mathcal{I}^\alpha$, $t(\ell r+1)$ rounds of Sherali–Adams certify that $|\mathcal{I}^\alpha(x)| \le \frac{3}{2}\epsilon$. We confirm that

$$\frac{m_\alpha}{m}\cdot n^{\lceil t/2\rceil/4\ell}\cdot\rho = \frac{\sqrt{pn^{k-t}}\cdot n^t}{pn^k}\cdot n^{\lceil t/2\rceil/4\ell}\cdot\rho \ll o(1),$$

whenever $t \ge 1$ and $\ell \ge 1$.

Finally, we handle α with $|\alpha| = t$ for $2 \le t$ and $pn^{k-t} < 1$. From Lemma 50 we have $\mathbf{E}\,|w_T| \ge \frac{1}{e}\log\frac{1}{1-pn^{k-t}}$, and with high probability, $|w_T| \le 4\log n$ for all $T \in [n]^t$. Taking $M = 4\log n$, we have that we meet the conditions of Propositions 45 and 49

$$M \le 4\log n \ll n^{\lceil k/2\rceil - \lceil t/2\rceil + \delta} = pn^{k-t}n^{\lfloor t/2\rfloor} = \sigma^2 n^{\lfloor t/2\rfloor},$$
$$\mathbf{E}\,|w_T| \ge \frac{1}{e}\log\frac{1}{1-pn^{k-t}} \ge \frac{1}{e}pn^{k-t} \gg \sqrt{\frac{pn^{k-t}\log n}{n^{\lfloor t/2\rfloor}}} = \sqrt{\frac{\sigma^2\log n}{n^{\lfloor t/2\rfloor}}},$$

where the last inequality is true whenever $pn^{k-t+\lfloor t/2\rfloor} = n^{\lceil k/2\rceil - \lceil t/2\rceil + \delta} \gg \log n$, which we have by assumption. So applying Propositions 45 and 49 to both $\frac{m}{m_\alpha}\mathcal{I}^\alpha$ and $-\frac{m}{m_\alpha}\mathcal{I}^\alpha$, we have that for

$$\rho = O\left(\frac{\sqrt{pn^{k-t}}\log n}{pn^{k-t}\sqrt{n^{\lfloor t/2\rfloor}}}\right) = O(\log n)\cdot\sqrt{\frac{1}{pn^{k-\lceil t/2\rceil}}} = O\left(\frac{\log n}{\sqrt{n^{\lceil k/2\rceil - \lceil t/2\rceil + \delta}}}\right),$$

so long as $\frac{m_\alpha}{m}n^{\lceil t/2\rceil/4\ell}\rho \le \frac{1}{2}\epsilon^{2\ell}$, $R = t(\ell r+1)$ rounds of Sherali–Adams certify that $|\mathcal{I}^\alpha(x)| \le \frac{3}{2}\epsilon$ with high probability. Verifying,

$$\frac{m_\alpha}{m}\cdot n^{\lceil t/2\rceil/4\ell}\cdot O\left(\frac{\log n}{\sqrt{n^{\lceil k/2\rceil - \lceil t/2\rceil + \delta}}}\right) = \frac{1}{\sqrt{pn^{k-t}}}\cdot n^{\lceil t/2\rceil/4\ell}\cdot O\left(\frac{\log n}{\sqrt{n^{\lceil k/2\rceil - \lceil t/2\rceil + \delta}}}\right),$$

which is maximized at $t = k$. By our choice of ℓ, the condition holds.

We therefore have (using Parseval's identity and $\|x\|_1 \leq \sqrt{k}\|x\|_2$ for $x \in \mathbb{R}^k$ to simplify Equation (21)) that the same number of rounds certifies that

$$\mathrm{OBJ}_{\mathcal{I}}(x) \leq \sum_{\alpha \subset [k]} \widehat{P}(\alpha) \cdot \mathcal{I}^{\alpha}(x) \leq \widehat{P}(\emptyset) + \sqrt{2^k}\frac{3}{2}\epsilon,$$

as desired. ◀

Using arguments analogous to the above along with the reasoning outlined in Theorem 4.9, proof 2 and Claim 6.7 from [2], we can also prove Theorem 40.

References

1 Mikhail Alekhnovich, Sanjeev Arora, and Iannis Tourlakis. Towards Strong Nonapproximability Results in the Lovász-Schrijver Hierarchy. *Computational Complexity*, 20(4):615–648, 2011. doi:10.1007/s00037-011-0027-z.

2 Sarah R. Allen, Ryan O'Donnell, and David Witmer. How to Refute a Random CSP. In Venkatesan Guruswami, editor, *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 689–708. IEEE Computer Society, 2015. doi:10.1109/FOCS.2015.48.

3 Sanjeev Arora, Béla Bollobás, László Lovász, and Iannis Tourlakis. Proving Integrality Gaps without Knowing the Linear Program. *Theory of Computing*, 2(2):19–51, 2006. doi:10.4086/toc.2006.v002a002.

4 Sanjeev Arora, László Lovász, Ilan Newman, Yuval Rabani, Yuri Rabinovich, and Santosh Vempala. Local Versus Global Properties of Metric Spaces. *SIAM J. Comput.*, 41(1):250–271, 2012. doi:10.1137/090780304.

5 Francisco Barahona and Ali Ridha Mahjoub. On the cut polytope. *Math. Programming*, 36(2):157–173, 1986. doi:10.1007/BF02592023.

6 Boaz Barak, Moritz Hardt, Thomas Holenstein, and David Steurer. Subsampling Mathematical Relaxations and Average-case Complexity. In Dana Randall, editor, *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, San Francisco, California, USA, January 23-25, 2011*, pages 512–531. SIAM, 2011. doi:10.1137/1.9781611973082.41.

7 Boaz Barak, Samuel B. Hopkins, Jonathan A. Kelner, Pravesh Kothari, Ankur Moitra, and Aaron Potechin. A Nearly Tight Sum-of-Squares Lower Bound for the Planted Clique Problem. In Irit Dinur, editor, *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 428–437. IEEE Computer Society, 2016. doi:10.1109/FOCS.2016.53.

8 Boaz Barak, Prasad Raghavendra, and David Steurer. Rounding Semidefinite Programming Hierarchies via Global Correlation. In Rafail Ostrovsky, editor, *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 472–481. IEEE Computer Society, 2011. doi:10.1109/FOCS.2011.95.

9 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.

10 Colin R Blyth. Expected Absolute Error of the Usual Estimator of the Binomial Parameter. *American Statistician*, pages 155–157, 1980.

11 Ravi Boppana. Eigenvalues and graph bisection: An average-case analysis. In *28th Annual Symposium on Foundations of Computer Science*, pages 280–285. IEEE, 1987.

12 Andrei Z. Broder, Alan M. Frieze, Stephen Suen, and Eli Upfal. Optimal construction of edge-disjoint paths in random graphs. *SIAM J. Comput.*, 28(2):541–573, 1999. doi:10.1137/S0097539795290805.

13 Siu On Chan, James R. Lee, Prasad Raghavendra, and David Steurer. Approximate Constraint Satisfaction Requires Large LP Relaxations. *J. ACM*, 63(4):34:1–34:22, October 2016. doi:10.1145/2811255.

14 M Charikar and A Wirth. Maximizing quadratic programs: extending Grothendieck's inequality. In *Foundations of Computer Science, 2004. Proceedings. 45th Annual IEEE Symposium on*, pages 54–60. IEEE, 2004.

15 Moses Charikar, Konstantin Makarychev, and Yury Makarychev. Integrality gaps for Sherali–Adams relaxations. In Michael Mitzenmacher, editor, *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pages 283–292. ACM, 2009. doi:10.1145/1536414.1536455.

16 Moses Charikar, Konstantin Makarychev, and Yury Makarychev. Local Global Tradeoffs in Metric Embeddings. *SIAM J. Comput.*, 39(6):2487–2512, 2010. doi:10.1137/070712080.

17 Nicholas Cook, Larry Goldstein, and Tobias Johnson. Size biased couplings and the spectral gap for random regular graphs. *Ann. Probab.*, 46(1):72–125, 2018. doi:10.1214/17-AOP1180.

18 Wenceslas Fernandez de la Vega and Claire Mathieu. Linear Programming Relaxations of Maxcut. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, pages 53–61, Philadelphia, PA, USA, 2007. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=1283383.1283390.

19 Charles Delorme and Svatopluk Poljak. Laplacian eigenvalues and the maximum cut problem. *Mathematical Programming*, 62(1-3):557–574, 1993.

20 Michel Marie Deza and Monique Laurent. *Geometry of cuts and metrics*, volume 15 of *Algorithms and Combinatorics*. Springer, Heidelberg, 1997. doi:10.1007/978-3-642-04295-9.

21 William E Donath and Alan J Hoffman. Lower bounds for the partitioning of graphs. In *Selected Papers Of Alan J Hoffman: With Commentary*, pages 437–442. World Scientific, 2003.

22 Uriel Feige and Robert Krauthgamer. The Probable Value of the Lovász–Schrijver Relaxations for Maximum Independent Set. *SIAM J. Comput.*, 32(2):345–370, 2003. doi:10.1137/S009753970240118X.

23 Uriel Feige and Michael Langberg. The RPR 2 rounding technique for semidefinite programs. In *International Colloquium on Automata, Languages, and Programming*, pages 213–224. Springer, 2001.

24 Uriel Feige and Eran Ofek. Spectral techniques applied to sparse random graphs. *Random Structures Algorithms*, 27(2):251–275, 2005. doi:10.1002/rsa.20089.

25 Miroslav Fiedler. Algebraic connectivity of graphs. *Czechoslovak mathematical journal*, 23(2):298–305, 1973.

26 Joel Friedman, Andreas Goerdt, and Michael Krivelevich. Recognizing More Unsatisfiable Random k-SAT Instances Efficiently. *SIAM J. Comput.*, 35(2):408–430, 2005. doi:10.1137/S009753970444096X.

27 Joel Friedman, Jeff Kahn, and Endre Szemerédi. On the second eigenvalue of random regular graphs. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing*, pages 587–598, 1989.

28 Alan Frieze and Ravi Kannan. The regularity lemma and approximation schemes for dense problems. In *Foundations of Computer Science, 1996. Proceedings., 37th Annual Symposium on*, pages 12–20. IEEE, 1996.

29 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.

30 Andreas Goerdt and Michael Krivelevich. Efficient Recognition of Random Unsatisfiable k-SAT Instances by Spectral Methods. In Afonso Ferreira and Horst Reichel, editors, *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. doi:10.1007/3-540-44693-1_26.

31 Dima Grigoriev. Linear lower bound on degrees of Positivstellensatz calculus proofs for the parity. *Theor. Comput. Sci.*, 259(1-2):613–622, 2001. doi:10.1016/S0304-3975(00)00157-2.

32 Johan Håstad. An NP-complete problem – some aspects of its solution and some possible applications. Master's thesis, Uppsala University, 1984.

33 Samuel B. Hopkins, Pravesh K. Kothari, Aaron Potechin, Prasad Raghavendra, Tselil Schramm, and David Steurer. The Power of Sum-of-Squares for Detecting Hidden Structures. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 720–731. IEEE Computer Society, 2017. doi:10.1109/FOCS.2017.72.

34 Pravesh 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 Symposium on Theory of Computing*, pages 132–145, 2017.

35 Pravesh K. Kothari, Raghu Meka, and Prasad Raghavendra. Approximating rectangles by juntas and weakly-exponential lower bounds for LP relaxations of CSPs. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 590–603, 2017. doi:10.1145/3055399.3055438.

36 David A. Levin and Yuval Peres. Counting walks and graph homomorphisms via Markov chains and importance sampling. *Amer. Math. Monthly*, 124(7):637–641, 2017. doi:10.4169/amer.math.monthly.124.7.637.

37 Bojan Mohar and Svatopluk Poljak. Eigenvalues in combinatorial optimization. In *Combinatorial and graph-theoretical problems in linear algebra*, pages 107–151. Springer, 1993.

38 Svatopluk Poljak. Polyhedral and eigenvalue approximations of the max-cut problem. In *Sets, graphs and numbers (Budapest, 1991)*, volume 60 of *Colloq. Math. Soc. János Bolyai*, pages 569–581. North-Holland, Amsterdam, 1992.

39 Svatopluk Poljak and Zsolt Tuza. The expected relative error of the polyhedral approximation of the max-cut problem. *Oper. Res. Lett.*, 16(4):191–198, 1994. doi:10.1016/0167-6377(94)90068-X.

40 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, Montreal, QC, Canada, June 19-23, 2017*, pages 121–131, 2017. doi:10.1145/3055399.3055417.

41 Prasad Raghavendra, Tselil Schramm, and David Steurer. High-dimensional estimation via sum-of-squares proofs. *CoRR*, abs/1807.11419, 2018. arXiv:1807.11419.

42 Prasad Raghavendra and Ning Tan. Approximating CSPs with global cardinality constraints using SDP hierarchies. In Yuval Rabani, editor, *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*, pages 373–387. SIAM, 2012. doi:10.1137/1.9781611973099.

43 Grant Schoenebeck. Linear Level Lasserre Lower Bounds for Certain k-CSPs. In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, October 25-28, 2008, Philadelphia, PA, USA*, pages 593–602. IEEE Computer Society, 2008. doi:10.1109/FOCS.2008.74.

44 Grant Schoenebeck, Luca Trevisan, and Madhur Tulsiani. Tight integrality gaps for Lovasz-Schrijver LP relaxations of vertex cover and max cut. In David S. Johnson and Uriel Feige, editors, *Proceedings of the 39th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, June 11-13, 2007*, pages 302–310. ACM, 2007. doi:10.1145/1250790.1250836.

45 Hanif D. Sherali and Warren P. Adams. A Hierarchy of Relaxations Between the Continuous and Convex Hull Representations for Zero-One Programming Problems. *SIAM J. Discrete Math.*, 3(3):411–430, 1990. doi:10.1137/0403036.

46 Konstantin Tikhomirov and Pierre Youssef. The spectral gap of dense random regular graphs. *arXiv preprint*, 2016. arXiv:1610.01765.

47 Luca Trevisan. Max Cut and the Smallest Eigenvalue. In *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing*, STOC '09, pages 263–272, New York, NY, USA, 2009. ACM. doi:10.1145/1536414.1536452.

48 Joel Tropp. An introduction to matrix concentration inequalities. *Foundations and Trendsin Machine Learning*, 8(1-2):1–230, 2015.

49 Yuichi Yoshida and Yuan Zhou. Approximation schemes via Sherali-Adams hierarchy for dense constraint satisfaction problems and assignment problems. In *Proceedings of the 5th conference on Innovations in theoretical computer science*, pages 423–438. ACM, 2014.

50 Uri Zwick. Outward Rotations: A Tool for Rounding Solutions of Semidefinite Programming Relaxations, with Applications to MAX CUT and Other Problems. In *Proceedings of the Thirty-first Annual ACM Symposium on Theory of Computing*, STOC '99, pages 679–687, New York, NY, USA, 1999. ACM. doi:10.1145/301250.301431.

A Characteristics of distributions of XOR-subformula coefficients

We now prove Lemma 50. We will use the following estimate for the mean absolte deviation of a binomial random variable.

▶ **Lemma 51** (e.g. [10]). *If X is distributed according to the binomial distribution $X \sim \mathrm{Bin}(n,p)$, then $\boldsymbol{E}\,|X - \boldsymbol{E}\,X| = \sqrt{\frac{2}{\pi}np(1-p)} + O(\frac{1}{\sqrt{n}})$.*

Proof of Lemma 50. We calculate the absolute value directly. Given that there are exactly k nonzero Y_t, the absolute value of X is distributed according to $|\mathrm{Bin}(k,\frac{1}{2}) - \frac{1}{2}k|$. Using the method of conditional expectations,

$$\mathbf{E}\,|X| = \sum_{k=0}^{N} \mathbf{Pr}\,\mathbb{I}[k \text{ nonzero } Y_t\text{'s}] \cdot \mathbf{E}\,|\mathrm{Bin}(k,\tfrac{t}{2}) - \tfrac{1}{2}k| \geq \sum_{k=0}^{N} \binom{N}{k} p^k (1-p)^{N-k} \cdot \sqrt{\frac{1}{2\pi}k},$$

where we have applied the estimate from Lemma 51. Letting $D(a\|b) = a\ln\frac{a}{b} + (1-a)\ln\frac{1-a}{1-b}$ be the relative entropy, we then have from Stirling's inequality that

$$\mathbf{E}\,|X| \geq \sum_{k=1}^{N} \sqrt{\frac{2\pi}{e^2}\frac{N}{k(N-k)}} \cdot \exp\left(-N \cdot D\left(\tfrac{k}{N}\|p\right)\right) \cdot \sqrt{\frac{1}{2\pi}k} \geq \frac{1}{e}\sum_{k=1}^{N} \exp\left(-N \cdot D\left(\tfrac{k}{N}\|p\right)\right), \quad (22)$$

Now, if $pN < 1$, we take

$$\textit{Equation (22)} \geq \frac{1}{e}\sum_{k=1}^{N} \exp\left(k\log\left(\frac{pN}{k}\right)\right) = \frac{1}{e}\sum_{k=1}^{N}\left(\frac{pN}{k}\right)^k \geq -\frac{1}{e}\log(1-pN) - O(p^N) \quad (23)$$

as desired.

If $pN \geq 1$, applying the change of variables $\ell = k - \lfloor pN \rfloor$ and $\delta = \frac{\ell}{N}$,

$$\textit{Equation (22)} \geq \frac{1}{e}\sum_{\substack{\ell=1-\lfloor pN\rfloor \\ \delta=\frac{\ell}{N}}}^{\lfloor(1-p)N\rfloor} \exp\left(-N \cdot D\left(p+\delta\|p\right)\right) \quad (24)$$

Now using the Taylor expansion for $\log(1-x)$ to simplify and restricting the sum over the range $\ell = [-\lfloor\sqrt{pN}\rfloor, \lfloor\sqrt{pN}\rfloor]$, we get the bound

$$\textit{Equation (24)} \geq \frac{1}{e} \sum_{\substack{\ell=-\lfloor\sqrt{pN}\rfloor \\ \delta=\frac{\ell}{N}}}^{\lfloor\sqrt{pN}\rfloor} \exp\left(-N\frac{\delta^2}{2}\right) \geq \frac{1}{e}\cdot 2\sqrt{pN}\cdot\exp(-\tfrac{p}{2}) \geq \frac{2}{e^{3/2}}\sqrt{pN}, \qquad (25)$$

as desired.

The first and second moment we can also obtain by calculation; the Z_t ensure that the summands have mean 0, and the Y_t give that the variance of the summands is p, which gives the result.

The tail bound $\mathbf{Pr}(|X| \geq (1+2t)\sqrt{pN}) \leq 2\exp(-t^2)$ comes from an application of Bernstein's inequality if $pN \geq 1$; when $pN < 1$, we again apply Bernstein's inequality, in which case we have

$$\mathbf{Pr}\left(|X| - \mathbf{E}\,|X| \geq s\right) \leq \exp\left(-\frac{1}{2}\frac{s^2}{pN + \frac{1}{3}s}\right),$$

and choosing $s = 1 + t$ gives the result. ◀

Typically-Correct Derandomization for Small Time and Space

William M. Hoza
Department of Computer Science, University of Texas at Austin, USA
https://williamhoza.com
whoza@utexas.edu

Abstract

Suppose a language L can be decided by a bounded-error randomized algorithm that runs in space S and time $n \cdot \text{poly}(S)$. We give a randomized algorithm for L that still runs in space $O(S)$ and time $n \cdot \text{poly}(S)$ that uses only $O(S)$ random bits; our algorithm has a low failure probability on all but a negligible fraction of inputs of each length. As an immediate corollary, there is a deterministic algorithm for L that runs in space $O(S)$ and succeeds on all but a negligible fraction of inputs of each length. We also give several other complexity-theoretic applications of our technique.

2012 ACM Subject Classification Theory of computation → Pseudorandomness and derandomization; Theory of computation → Probabilistic computation; Theory of computation → Complexity classes

Keywords and phrases Derandomization, pseudorandomness, space complexity

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.9

Funding *William M. Hoza*: Supported by the NSF GRFP under Grant DGE-1610403 and by a Harrington Fellowship from UT Austin.

Acknowledgements We thank Michael Forbes, Scott Aaronson, David Zuckerman, Adam Klivans, and Anna Gál for helpful comments on an early draft of this paper. We thank Amnon Ta-Shma, Lijie Chen, Chris Umans, David Zuckerman, Adam Klivans, Anna Gál, Gil Cohen, Shachar Lovett, Oded Goldreich, and Avi Wigderson for helpful discussions.

1 Introduction

1.1 The Power of Randomness When Time and Space Are Limited

A central goal of complexity theory is to understand the relationship between three fundamental resources: time, space, and randomness. Based on a long line of research [42, 9, 6, 29, 20, 37, 25], most complexity theorists believe that randomized decision algorithms can be made deterministic without paying too much in terms of time and space. Specifically, suppose a language L can be decided by a randomized algorithm that runs in time $T = T(n) \geq n$ and space $S = S(n) \geq \log n$. Klivans and van Melkebeek showed that assuming some language in $\mathbf{DSPACE}(n)$ has exponential circuit complexity, there is a deterministic algorithm for L that runs in time $\text{poly}(T)$ and space $O(S)$ [25].[1]

Proving the hypothesized circuit lower bound seems unlikely for the foreseeable future. In the 90s and early 2000s, researchers managed to prove powerful *unconditional* derandomization theorems by focusing on the space complexity of the deterministic algorithm. For example, Nisan and Zuckerman showed that if $S \geq T^{\Omega(1)}$, there is a deterministic algorithm for

[1] More generally, Klivans and van Melkebeek constructed a pseudorandom generator that fools size-T circuits on T input bits under this assumption. The generator has seed length $O(\log T)$ and is computable in $O(\log T)$ space.

L that runs in space $O(S)$ [30].[2] Alas, in the past couple of decades, progress on such general, unconditional derandomization has stalled. Nobody has managed to extend the Nisan-Zuckerman theorem to a larger regime of pairs (T, S), and researchers have been forced to focus on more restricted models of computation.

In this paper, we focus on *highly efficient* randomized algorithms. That is, we consider the case that T and S are both small, such as $T \leq \widetilde{O}(n)$ and $S \leq O(\log n)$.

1.2 Our Results

1.2.1 Reducing the Amount of Randomness to $O(S)$

Suppose $T \leq n \cdot \text{poly}(S)$. For our main result, we give a randomized algorithm for L that still runs in time $n \cdot \text{poly}(S)$ and space $O(S)$ that uses only $O(S)$ random bits. The catch is that our algorithm is only guaranteed to succeed on *most* inputs. The fraction of "bad" inputs of length n is at most 2^{-S^c}, where $c \in \mathbb{N}$ is an arbitrarily large constant. On "good" inputs, our algorithm's failure probability is at most $2^{-S^{1-\alpha}}$, where $\alpha > 0$ is an arbitrarily small constant.

1.2.2 Eliminating Randomness Entirely

From the result described in the preceding paragraph, a deterministic algorithm that runs in space $O(S)$ follows immediately by iterating over all $O(S)$-bit random strings. We can express this theorem in terms of complexity classes using terminology introduced by Kinne et al. for typically-correct algorithms [24]. Suppose L is a language, $\mathbf{C}$ is a complexity class, and $\varepsilon(n)$ is a function. We say that L *is within* ε *of* $\mathbf{C}$ if there is some $L' \in \mathbf{C}$ such that for every n,

$$\Pr_{x \in \{0,1\}^n}[x \in L \Delta L'] \leq \varepsilon(n). \tag{1}$$

If $\mathbf{C}$ and $\mathbf{C}'$ are complexity classes, we say that $\mathbf{C}$ *is within* ε *of* $\mathbf{C}'$ if every language in $\mathbf{C}$ is within ε of $\mathbf{C}'$. In these terms, our result is that

$$\mathbf{BPTISP}(n \cdot \text{poly}(S), S) \text{ is within } 2^{-S^c} \text{ of } \mathbf{DSPACE}(S). \tag{2}$$

Here, $\mathbf{BPTISP}(T, S)$ is the class of languages that can be decided by a bounded-error randomized algorithm that runs in time $O(T(n))$ and space $O(S(n))$, and $\mathbf{DSPACE}(S)$ is the class of languages that can be decided by a deterministic algorithm that runs in space $O(S)$. Note that if $S \geq n^{\Omega(1)}$, the mistake rate in Equation (2) drops below 2^{-n}. Since there are only 2^n inputs of length n, the algorithm must in fact be correct on all inputs. Our result can therefore be viewed as a generalization of the Nisan-Zuckerman theorem $\mathbf{BPTISP}(\text{poly}(S), S) \subseteq \mathbf{DSPACE}(S)$ [30].

1.2.3 Derandomization with Advice

Adleman's argument [1] shows that $\mathbf{BPL} \subseteq \mathbf{L}/\text{poly}$. We study the problem of derandomizing $\mathbf{BPL}$ with as little advice as possible. Goldreich and Wigderson discovered a critical threshold: roughly, if an algorithm can be derandomized with fewer than n bits of advice, then there is a *typically-correct* derandomization of the algorithm with *no* advice [15].[3]

---

[2] More generally, the Nisan-Zuckerman theorem applies as long as the original randomized algorithm for L uses only $\text{poly}(S)$ random bits, regardless of how much time it takes.

[3] This result also requires that (a) most advice strings are "good", and (b) there is an appropriate efficient extractor.

Motivated by this phenomenon, Fortnow and Klivans proved that $\mathbf{BPL} \subseteq \mathbf{L}/O(n)$ [12]. We refine their argument and show that $\mathbf{BPL} \subseteq \mathbf{L}/(n + O(\log^2 n))$, getting very near the critical threshold of n bits of advice. More interestingly, we show that the connection identified by Goldreich and Wigderson [15] works the other way: in the space-bounded setting, typically-correct derandomizations imply derandomizations with just a little advice. Combining with our main result gives that for every constant $c \in \mathbb{N}$,

$$\mathbf{BPTISP}(\widetilde{O}(n), \log n) \subseteq \mathbf{L}/(n - \log^c n). \tag{3}$$

1.2.4 Derandomizing Turing Machines

All algorithms in the results mentioned so far are formulated in a general *random-access* model, i.e., the algorithm can read any specified bit of its input in a single step. (See Section 2.2 for details.) We also study the weaker *multitape Turing machine* model. The main weakness of the Turing machine model is that if its read head is at position i of its input and it wishes to read bit j of its input, it must spend $|i - j|$ steps moving its read head to the appropriate location. Let $\mathbf{BPTISP}_{\mathrm{TM}}(T, S)$ denote the class of languages that can be decided by a bounded-error randomized Turing machine that runs in time $O(T(n))$ and space $O(S(n))$.

1.2.4.1 Beyond Linear Advice

We give a typically-correct derandomization for $\mathbf{BPTISP}_{\mathrm{TM}}$ analogous to our main result but with a lower mistake rate. In terms of advice, our derandomization implies that for every constant $c \in \mathbb{N}$,

$$\mathbf{BPTISP}_{\mathrm{TM}}(\widetilde{O}(n), \log n) \subseteq \mathbf{L}/O\left(\frac{n}{\log^c n}\right). \tag{4}$$

Equation (4) gives an interesting example of a class of **BPL** algorithms that can be derandomized with $o(n)$ bits of advice.

1.2.4.2 Beyond Quasilinear Time

Using different techniques, we also show how to derandomize log-space Turing machines that use almost a *quadratic* amount of time. In particular, we show that if $TS^2 \leq o(n^2/\log n)$, then

$$\mathbf{BPTISP}_{\mathrm{TM}}(T, S) \text{ is within } o(1) \text{ of } \mathbf{DTISP}(\mathrm{poly}(n), S). \tag{5}$$

1.2.5 Disambiguating Nondeterministic Algorithms

For some of our derandomization results, we give analogous theorems regarding *unambiguous* simulations of *nondeterministic* algorithms. We defer a discussion of these results to Section 6.

1.3 Techniques

1.3.1 "Out of Sight, out of Mind"

Our typically-correct derandomizations work by treating the *input as a source of randomness*. This idea was pioneered by Goldreich and Wigderson [15]. For the sake of discussion, let $\mathcal{A}$ be a randomized algorithm that uses n random bits. A naïve strategy for derandomizing $\mathcal{A}$

is to run $\mathcal{A}(x, x)$. Most random strings of $\mathcal{A}$ lead to the right answer, so it is tempting to think that for most x, $\mathcal{A}(x, x)$ will give the right answer. This reasoning is flawed, because $\mathcal{A}$ might behave poorly when its input is *correlated* with its random bits.

In this work, we avoid these troublesome correlations using a simple idea embodied by the adage "out of sight, out of mind." We use *part* of the input as a source of randomness while $\mathcal{A}$ is processing *the rest* of the input.

To go into more detail, suppose $\mathcal{A}$ runs in time $\widetilde{O}(n)$ and space $O(\log n)$. Our randomness-efficient simulation of $\mathcal{A}$ operates in $\operatorname{polylog}(n)$ *phases*. At the beginning of a new phase, we pick a random $\operatorname{polylog}(n)$-bit block $x|_I$ of the input x. We apply a seeded extractor to $x|_I$, giving a string of length $\Theta(\log^2 n)$. We apply Nisan's pseudorandom generator for space-bounded computation [26], giving a pseudorandom string of length $\widetilde{O}(n)$. We use the pseudorandom string to run the simulation of $\mathcal{A}$ forward until it tries to read from $x|_I$, at which time we pause the simulation of $\mathcal{A}$ and move on to the next phase.

The key point is that the output of the extractor is processed without ever looking at $x|_I$, the input to the extractor. Extractors are good *samplers* [44], and $\mathcal{A}$ only has polynomially many possible configurations, so for most x, the output of the extractor is essentially as good as a uniform random seed to Nisan's generator. Therefore, in each phase, with high probability, we successfully simulate $n/\operatorname{polylog}(n)$ steps of $\mathcal{A}$ before it reads from $x|_I$ and we have to move on to the next phase. Thus, with high probability, after $\operatorname{polylog}(n)$ phases, the simulation of $\mathcal{A}$ is complete.

Each bit of the output of Nisan's generator can be computed in time[4] $\operatorname{polylog}(n)$ and space $O(\log n)$. Therefore, our simulation of $\mathcal{A}$ still runs in time $\widetilde{O}(n)$ and space $O(\log n)$, but now it uses just $\operatorname{polylog}(n)$ random bits ($O(\log n)$ random bits per phase to pick the random block I and to pick a seed for the extractor).

The reader may wonder whether we could have achieved the same effect by simply directly applying Nisan's generator from the start – its seed length is $\operatorname{polylog}(n)$, after all. The point is that Nisan's generator requires *two-way* access to its seed, whereas our simulation only uses one-way access to its random bits. During our simulation, we are able to give Nisan's generator two-way access to its seed, because we have two-way access to the *input* x from which we extract that seed.

Finally, because our simulation reads its $\operatorname{polylog}(n)$ random bits from left to right, we can further reduce the number of random bits to just $O(\log n)$ by applying the Nisan-Zuckerman pseudorandom generator [30].

1.3.2 Other Techniques

Our derandomizations with advice are based on Fortnow and Klivans' technique for proving $\mathbf{BPL} \subseteq \mathbf{L}/O(n)$ [12] and Nisan's technique for proving $\mathbf{RL} \subseteq \mathbf{SC}$ [28]. Our derandomization of $\mathbf{BPTISP}_{\mathrm{TM}}$ with a low mistake rate uses a similar "out of sight, out of mind" technique as our main result. The lower mistake rate is achieved by exploiting knowledge of the region of the input that will be processed in the near future, based on the locality of the Turing machine's read head. Our derandomization of $\mathbf{BPTISP}_{\mathrm{TM}}(T, S)$ for $T(n) \approx n^2$ is based on a seed-extending pseudorandom generator for multiparty communication protocols by Kinne et al. [24].

[4] See work by Diehl and van Melkebeek [11] for an even faster implementation of Nisan's generator.

1.4 Related Work

We will only mention some highlights of the large body of research on unconditional derandomization of time- and space-bounded computation. Fix $L \in \mathbf{BPTISP}(T, S)$. Nisan gave a randomized algorithm for L that runs in time $\mathrm{poly}(T)$ and space $O(S \log T)$ that uses only $O(S \log T)$ random bits [26]. Nisan also gave a deterministic algorithm for L that runs in time $2^{O(S)}$ and space $O(S \log T)$ [28]. Nisan and Zuckerman gave a randomized algorithm for L that runs in time $\mathrm{poly}(T)$ and space $O(S + T^{\varepsilon})$ that uses only $O(S + T^{\varepsilon})$ random bits, where $\varepsilon > 0$ is an arbitrarily small constant [30] (this is a generalization of the result mentioned in Section 1.1). Saks and Zhou gave a deterministic algorithm for L that runs in space $O(S\sqrt{\log T})$ [32]. Combining the techniques from several of these works, Armoni [4] gave a deterministic algorithm for L that runs in space[5]

$$O\left(S \cdot \sqrt{\frac{\log T}{\max\{1, \log S - \log\log T\}}}\right). \tag{6}$$

Armoni's algorithm remains the most space-efficient derandomization known for all T and S. When $T = \widetilde{\Theta}(n)$ and $S = \Theta(\log n)$, Armoni's algorithm runs in space $\Theta(\log^{3/2} n)$, just like the earlier Saks-Zhou algorithm [32]. Cai et al. gave a time-space tradeoff [10] interpolating between Nisan's deterministic algorithm [28] and the Saks-Zhou algorithm [32].

All of the preceding results apply, *mutatis mutandis*, to derandomizing algorithms that use at most T random bits, regardless of how much time they take. In contrast, our proofs crucially rely on the fact that a time-T algorithm queries its *input* at most T times. This aspect of our work is shared by work by Beame et al. [8] on time-space lower bounds.

Goldreich and Wigderson's idea of using the input as a source of randomness for a typically-correct derandomization [15] has been applied and developed by several researchers [5, 41, 23, 43, 35, 24, 33, 3]; see related survey articles by Shaltiel [34] and by Hemaspaandra and Williams [19]. Researchers have proven unconditional typically-correct derandomization results for several restricted models, including sublinear-time algorithms [43, 35], communication protocols [35, 24], constant-depth circuits [35, 24], and streaming algorithms [35]. On the other hand, Kinne et al. proved that any typically-correct derandomization of **BPP** with a sufficiently low mistake rate would imply strong circuit lower bounds [24]. We are the first to study typically-correct derandomization for algorithms with simultaneous bounds on time and space.

1.5 Outline of This Paper

In Section 2, we discuss random-access models of computation and extractors. In Section 3, we give our derandomization of $\mathbf{BPTISP}(n \cdot \mathrm{poly}(S), S)$. In Section 4, we give our two derandomizations of $\mathbf{BPTISP}_{\mathrm{TM}}(T, S)$. In Section 5, we discuss derandomization with advice. Section 6 concerns disambiguation of nondeterministic algorithms, and we conclude in Section 7 with some suggested directions for further research.

[5] Actually, the space bound given in Equation (6) is achieved by using better extractors than were known when Armoni wrote his paper [4, 22].

2 Preliminaries

2.1 General Notation

Strings

For strings x, y, let $x \circ y$ denote the concatenation of x with y. For a natural number n, let $[n] = \{1, 2, \ldots, n\}$. For a string $x \in \{0,1\}^n$ and a set $I = \{i_1 < i_2 < \cdots < i_\ell\} \subseteq [n]$, let $x|_I = x_{i_1} x_{i_2} \ldots x_{i_\ell} \in \{0,1\}^\ell$.

Sets

For a finite set X, we will use the notations $\#X$ and $|X|$ interchangeably to refer to the number of elements of X. For $X \subseteq \{0,1\}^n$, let $\text{density}(X) = |X|/2^n$. We will sometimes omit the parentheses, e.g., $\text{density}\{000, 111\} = 0.25$. We identify a language $L \subseteq \{0,1\}^*$ with its indicator function $L : \{0,1\}^* \to \{0,1\}$, i.e.,

$$L(x) = \begin{cases} 1 & \text{if } x \in L \\ 0 & \text{if } x \notin L. \end{cases} \tag{7}$$

Probability

If X and Y are probability distributions on the same space, we write $X \sim_\varepsilon Y$ to indicate that X and Y are ε-close in total variation distance. For $T \in \mathbb{N}$, let U_T denote the uniform distribution over $\{0,1\}^T$.

2.2 Random-Access Algorithms

Our main theorems govern general *random-access algorithms*. Our results are not sensitive to the specific choice of model of random-access computation. For concreteness, following Fortnow and van Melkebeek [13], we will work with the *random-access Turing machine* model. This model is defined like the standard multitape Turing machine model, except that each ordinary tape is supplemented with an "index tape" that can be used to move the ordinary tape's head to an arbitrary specified location in a single step. See the paper by Fortnow and van Melkebeek [13] for details.

A *randomized random-access Turing machine* is a random-access Turing machine equipped with an additional read-only tape, initialized with random bits, that can only be read from *left to right*. Thus, if the algorithm wishes to reread old random bits, it needs to have copied them to a work tape, which counts toward the algorithm's space usage. The random tape does not have a corresponding index tape.

For functions $T : \mathbb{N} \to \mathbb{N}$ and $S : \mathbb{N} \to \mathbb{N}$, we define $\mathbf{BPTISP}(T, S)$ to be the class of languages L such that there is a randomized random-access Turing machine $\mathcal{A}$ such that on input $x \in \{0,1\}^n$, $\mathcal{A}(x)$ always halts in time $O(T(n))$, $\mathcal{A}(x)$ always touches $O(S(n))$ cells on all of its read-write tapes, and $\Pr[\mathcal{A}(x) = L(x)] \geq 2/3$.

2.3 Randomized Branching Programs

Our algorithms are most naturally formulated in terms of branching programs, a standard *nonuniform* model of time- and space-bounded computation. Recall that in a digraph, a *terminal vertex* is a vertex with no outgoing edges. In the following definition, n is the number of input bits and m is the number of random bits.

▶ **Definition 1.** *A* randomized branching program *on* $\{0,1\}^n \times \{0,1\}^m$ *is a directed acyclic graph, where each nonterminal vertex* v *is labeled with two indices* $i(v) \in [n], j(v) \in [m]$ *and has four outgoing edges labeled with the four two-bit strings. If* $\mathcal{P}$ *is a randomized branching program, we let* $V(\mathcal{P})$ *be the set of vertices of* $\mathcal{P}$*.*

The interpretation is that from vertex v, the program follows the edge labeled $x_{i(v)}y_{j(v)}$, where x is the input and y is the random string. This interpretation is formalized by the following definition, which sets $\mathcal{P}(v;x,y)$ to be the vertex reached from v on input x using randomness y.

▶ **Definition 2.** *Suppose* $\mathcal{P}$ *is a randomized branching program on* $\{0,1\}^n \times \{0,1\}^m$*. We identify* $\mathcal{P}$ *with a function* $\mathcal{P} : V(\mathcal{P}) \times \{0,1\}^n \times \{0,1\}^m \to V(\mathcal{P})$ *defined as follows. Fix* $v \in V(\mathcal{P}), x \in \{0,1\}^n, y \in \{0,1\}^m$*. Take a walk through* $\mathcal{P}$ *by starting at* v *and, having reached vertex* u*, following the edge labeled* $x_{i(u)}y_{j(u)}$*. Then* $\mathcal{P}(v;x,y)$ *is the terminal vertex reached by this walk.*

As previously discussed, random-access Turing machines can only access their random bits from left to right. This corresponds to an *R-OW randomized branching program.*

▶ **Definition 3.** *An* R-OW randomized branching program *is a randomized branching program* $\mathcal{P}$ *such that for every edge* (v,v') *between two nonterminal vertices,* $j(v') \in \{j(v), j(v)+1\}$*.*

The term "R-OW" indicates that the branching program has "random access" to its input bits and "one-way access" to its random bits.

The *size* of a branching program is defined as $\text{size}(\mathcal{P}) = |V(\mathcal{P})|$. The *length* of the program, $\text{length}(\mathcal{P})$, is defined to be the length of the longest path through the program. Observe that $\mathbf{BPTISP}(T,S)$ corresponds to R-OW randomized branching programs of size $2^{O(S)}$ and length $O(T)$.

Many of our algorithms will use a *restriction* operation that we now introduce.

▶ **Definition 4.** *Suppose* $\mathcal{P}$ *is a randomized branching program on* $\{0,1\}^n$ *and* $I \subseteq [n]$*. Let* $\mathcal{P}|_I$ *be the program obtained from* $\mathcal{P}$ *by deleting all outgoing edges from vertices* v *such that* $i(v) \notin I$*.*

So in $\mathcal{P}|_I$, there are two types of terminal vertices: vertices that were terminal in $\mathcal{P}$, and vertices v that are now terminal because $i(v) \notin I$. The computation $\mathcal{P}|_I(v;x,y)$ halts when it reaches either type of terminal vertex. Thus, $\mathcal{P}|_I(v;x,y)$ does not depend on $x|_{[n]\setminus I}$, because $\mathcal{P}|_I(v;x,y)$ outputs the vertex reached by running the computation $\mathcal{P}(v;x,y)$ until it finishes or it tries to read from $x|_{[n]\setminus I}$.

2.4 Extractors

Recall that a (k,ε)-*extractor* is a function $\mathsf{Ext} : \{0,1\}^\ell \times \{0,1\}^d \to \{0,1\}^s$ such that if X has "min-entropy" at least k and $Y \sim U_d$ is independent of X, then $\mathsf{Ext}(X,Y) \sim_\varepsilon U_s$. It can be shown nonconstructively that for every ℓ, k, ε, there exists Ext with $d \leq \log(\ell - k) + 2\log(1/\varepsilon) + O(1)$ and $s \geq k + d - 2\log(1/\varepsilon) - O(1)$ (see, e.g., Vadhan's monograph [38]).

We will need a computationally efficient extractor. The extractor literature has mainly focused on the time complexity of computing extractors, but we are concerned with space complexity, too. This paper is not meant to be about extractor constructions, so we encourage the reader to simply pretend that optimal extractors can be computed in a single step with no space overhead. In actuality, we will use two incomparable non-optimal extractors.

To prove our main results, we will use an extractor by Shaltiel and Umans [36]. The benefit of the Shaltiel-Umans extractor is that it allows for small error ε.

▶ **Theorem 5** ([36]). *Fix a constant $\alpha > 0$. For every $\ell, k \in \mathbb{N}, \varepsilon > 0$ such that $k \geq \log^{4/\alpha} \ell$ and $k \geq \log^{4/\alpha}(1/\varepsilon)$, there is a (k,ε)-extractor $\mathsf{SUExt} : \{0,1\}^\ell \times \{0,1\}^d \to \{0,1\}^s$ where $d \leq O\left(\log \ell + \frac{\log \ell \log(1/\varepsilon)}{\log k}\right)$ and $s \geq k^{1-\alpha}$. Given x, y, k, and ε, $\mathsf{SUExt}(x,y)$ can be computed in time $\mathrm{poly}(\ell)$ and space $O(d)$.*

To derandomize **BPL** with as little advice as possible, we will use an extractor by Guruswami, Umans, and Vadhan [16] (not the most famous extractor from their work, but a slight variant). The benefit of the GUV extractor is that it outputs a constant fraction of the entropy.

▶ **Theorem 6** ([16]). *Let $\alpha, \varepsilon > 0$ be constant. For every $\ell, k \in \mathbb{N}$, there is a (k,ε)-extractor $\mathsf{GUVExt} : \{0,1\}^\ell \times \{0,1\}^d \to \{0,1\}^s$ with $s \geq (1-\alpha)k$ and $d \leq O(\log \ell)$ such that given x and y, $\mathsf{GUVExt}(x,y)$ can be computed in $O(\log \ell)$ space.*

In both cases, the original authors [36, 16] did not explicitly analyze the space complexity of their extractors, so we explain in Appendices A and B why these extractors can be implemented in small space. (We remark that Hartman and Raz also constructed small-space extractors [18], but the seed lengths of their extractors are too large for us.)

2.4.1 Extractors as Samplers

We will actually only be using extractors for their *sampling* properties. The connection between extractors and samplers was first discovered by Zuckerman [44]. The following standard proposition expresses this connection for non-Boolean functions.

▶ **Proposition 7** ([44]). *Suppose $\mathsf{Ext} : \{0,1\}^\ell \times \{0,1\}^d \to \{0,1\}^s$ is a (k,ε)-extractor and $f : \{0,1\}^s \to V$ is a function. Let $\delta = \varepsilon|V|/2$. Then*

$$\#\{x \in \{0,1\}^\ell : f(U_s) \not\approx_\delta f(\mathsf{Ext}(x, U_d))\} \leq 2^{k+1}|V|. \tag{8}$$

For completeness, we include a proof of Proposition 7 in Appendix C, since the specific statement of Proposition 7 does not appear in Zuckerman's paper [44].

2.5 Constructibility

We say that $f : \mathbb{N} \to \mathbb{N}$ is *constructible* in space $S(n)$, time $T(n)$, etc. if there is a deterministic random-access Turing machine $\mathcal{A}$ that runs in the specified resource bounds with $\mathcal{A}(1^n) = f(n)$, written in binary. As usual, we say that f is *space constructible* if f is constructible in space $O(f(n))$. We say that $\delta : \mathbb{N} \to [0,1]$ is constructible in specified resource bounds if δ can be written as $\delta(n) = \frac{\delta_1(n)}{\delta_2(n)}$, where $\delta_1, \delta_2 : \mathbb{N} \to \mathbb{N}$ are both constructible in the specified resource bounds.

3 Derandomizing Efficient Random-Access Algorithms

3.1 Main Technical Algorithm: Low-Randomness Simulation of Branching Programs

Suppose $\mathcal{P}$ is an R-OW randomized branching program on $\{0,1\}^n \times \{0,1\}^T$ of length T and size 2^S. (As a reminder, such a program models $\mathbf{BPTISP}(T,S)$.) Given $\mathcal{P}$, v_0, and x, the distribution $\mathcal{P}(v_0; x, U_T)$ can trivially be sampled in time $T \cdot \mathrm{poly}(S)$ and space $O(S)$ using T random bits. Our main technical result is an efficient typically-correct algorithm for approximately sampling $\mathcal{P}(v_0; x, U_T)$ using roughly T/n random bits.

▶ **Theorem 8.** *For each constant $c \in \mathbb{N}$, there is a randomized algorithm A with the following properties. Suppose $\mathcal{P}$ is an R-OW randomized branching program on $\{0,1\}^n \times \{0,1\}^T$ with $S \geq \log n$, where $S \stackrel{\text{def}}{=} \lceil \log \text{size}(\mathcal{P}) \rceil$. Suppose $v_0 \in V(\mathcal{P})$, $T \geq \text{length}(\mathcal{P})$, and $x \in \{0,1\}^n$. Then $\mathsf{A}(\mathcal{P}, v_0, x, T)$ outputs a vertex $v \in V(\mathcal{P})$ in time[6] $T \cdot \text{poly}(S)$ and space $O(S)$ using $\lceil T/n \rceil \cdot \text{poly}(S)$ random bits. Finally, for every such $\mathcal{P}, v_0, T$,*

$$\text{density}\{x \in \{0,1\}^n : \mathsf{A}(\mathcal{P}, v_0, x, T) \not\sim_{\exp(-cS)} \mathcal{P}(v_0; x, U_T)\} \leq 2^{-S^c}. \tag{9}$$

The algorithm of Theorem 8 relies on Nisan's pseudorandom generator [26]. The seed length of Nisan's generator is not $O(S)$, but Nisan's generator does *run* in space $O(S)$, given two-way access to the seed.

▶ **Theorem 9** ([26])**.** *For every $S, T \in \mathbb{N}, \varepsilon > 0$ with $T \leq 2^S$, there is a generator $\mathsf{NisGen} : \{0,1\}^s \to \{0,1\}^T$ with seed length $s \leq O((S + \log(1/\varepsilon)) \cdot \log T)$, such that if $\mathcal{P}$ is an R-OW randomized branching program of size 2^S, v is a vertex, and x is an input, then*

$$\mathcal{P}(v; x, \mathsf{NisGen}(U_s)) \sim_\varepsilon \mathcal{P}(v; x, U_T). \tag{10}$$

Given S, T, ε, z, i, the bit $\mathsf{NisGen}(z)_i$ can be computed in time $\text{poly}(S, \log(1/\varepsilon))$ and space $O(S + \log(1/\varepsilon))$.

Algorithm 1: The algorithm A of Theorem 8.

```
if S^{c+1} > ⌊n/9⌋ then
    Directly simulate P(v_0; x, U_T) using T random bits
else
    Let I_1, I_2, ..., I_B ⊆ [n] be disjoint sets of size S^{c+1} with B as large as possible
    Initialize v ← v_0
    repeat r times                          /* r is given by Equation (11) */
        Pick b ∈ [B] uniformly at random and let I ← I_b
        Pick y ∈ {0,1}^{O(S)} uniformly at random
        Update v ← P|_{[n]\I}(v; x, NisGen(SUExt(x|_I, y)))
    end
    return v
end
```

For Theorem 8, we can replace T with $\min\{T, 2^S\}$ without loss of generality, so we will assume that $T \leq 2^S$. The algorithm A is given in Algorithm 1.

Parameters

Set

$$r \stackrel{\text{def}}{=} \max\left\{\left\lceil \frac{8T}{B-8} \right\rceil, 8(cS+1)\right\} = \lceil T/n \rceil \cdot \text{poly}(S). \tag{11}$$

The parameter r is the number of "phases" of A as outlined in Section 1.3.1. Note that if $S^{c+1} \leq \lfloor n/9 \rfloor$, then $B \geq 9$, so Equation (11) makes sense. Naturally, Nisan's generator

[6] The graph of $\mathcal{P}$ should be encoded in adjacency list format, so that the neighborhood of a vertex v can be computed in $\text{poly}(S)$ time.

NisGen is instantiated with the parameters S, T from the statement of Theorem 8. The error of NisGen is set at

$$\varepsilon \stackrel{\text{def}}{=} \frac{e^{-cS}}{4r} = 2^{-\Theta(S)}. \tag{12}$$

That way, the seed length of NisGen is $s \leq O(S \log T) \leq O(S^2)$. The algorithm A also relies on the Shaltiel-Umans extractor SUExt of Theorem 5. This extractor is instantiated with source length $\ell \stackrel{\text{def}}{=} S^{c+1}$, $\alpha \stackrel{\text{def}}{=} 2/3$, error

$$\varepsilon' \stackrel{\text{def}}{=} \frac{e^{-cS}}{2r \cdot 2^S} = 2^{-\Theta(S)}, \tag{13}$$

and entropy

$$k \stackrel{\text{def}}{=} \max\{s^3, \log^6 \ell, \log^6(1/\varepsilon)\} = \Theta(S^6). \tag{14}$$

Our choice of k explicitly meets the hypotheses of Theorem 5, and by construction, $k^{1-\alpha} \geq s$, so we can think of SUExt as outputting s bits.

Efficiency

We now analyze the computational efficiency of A. First, we bound the running time. If $S^{c+1} > \lfloor n/9 \rfloor$, then A clearly runs in time $T \cdot \text{poly}(S)$. Otherwise, A repeatedly replaces v with one of its neighbors a total of at most T times, since $T \geq \text{length}(\mathcal{P})$. Each such step requires computing a bit of Nisan's generator, which takes time $\text{poly}(S)$, times $\text{poly}(S)$ steps to compute each bit of the seed of Nisan's generator by running SUExt. Thus, overall, A runs in time $T \cdot \text{poly}(S)$.

Next, we bound the space complexity of A. If $S^{c+1} > \lfloor n/9 \rfloor$, then A clearly runs in space $O(S + \log T) = O(S)$. Otherwise, space is required to store a loop index ($O(\log r)$ bits), the vertex v ($O(S)$ bits), the index b ($O(\log n)$ bits), and the seed y ($O(S)$ bits). These terms are all bounded by $O(S)$. Running SUExt takes $O(\log \ell + \frac{\log \ell \log(1/\varepsilon')}{\log k})$ bits of space. Since $k \geq S^{\Omega(1)}$, $\frac{\log \ell}{\log k} \leq O(1)$, and hence the space used for SUExt is only $O(\log S + \log(1/\varepsilon')) = O(S)$. Finally, running NisGen takes $O(S + \log(1/\varepsilon)) = O(S)$ bits of space. Therefore, overall, A runs in space $O(S)$.

Finally, we bound the number of random bits used by A. If $S^{c+1} > \lfloor n/9 \rfloor$, then A uses T random bits, which is at most $\frac{9T(1+S^{c+1})}{n}$ in this case. Otherwise, in each iteration of the loop, A uses $O(\log n)$ random bits for b, plus $O(S)$ random bits for y. Therefore, overall, the number of random bits used by A is $O(rS)$, which is $\lceil T/n \rceil \cdot \text{poly}(S)$.

Correctness

We now turn to proving Equation (9). If $S^{c+1} > \lfloor n/9 \rfloor$, then obviously $\mathsf{A}(\mathcal{P}, v_0, x, T) \sim \mathcal{P}(v_0; x, U_T)$. Assume, therefore, that $S^{c+1} \leq \lfloor n/9 \rfloor$. The proof will be by a hybrid argument with three hybrid distributions. The first hybrid distribution is defined by the algorithm H_1 given by Algorithm 2.

We need a standard fact about Markov chains. Suppose M and M' are stochastic matrices (i.e., each row is a probability vector) of the same size. We write $M \sim_\gamma M'$ to mean that for each row index i, the probability distributions M_i and M'_i are γ-close in total variation distance.

▶ **Lemma 10.** *If $M \sim_\gamma M'$, then $M^r \sim_{\gamma r} (M')^r$.*

Algorithm 2: The algorithm H_1 defining the first hybrid distribution used to prove Equation (9). The only difference between A and H_1 is that H_1 picks a uniform random seed for NisGen, instead of extracting the seed from the input.

Initialize $v \leftarrow v_0$
repeat r **times**
  Pick $b \in [B]$ uniformly at random and let $I \leftarrow I_b$
  Pick $y' \in \{0,1\}^s$ uniformly at random
  Update $v \leftarrow \mathcal{P}|_{[n]\setminus I}(v; x, \mathsf{NisGen}(y'))$
end
return v

For a proof of Lemma 10, see, e.g., work by Saks and Zhou [32, Proposition 2.3]. We are now ready to prove that for most x, the behavior of A is statistically similar to the behavior of H_1.

▷ Claim 11 ($\mathsf{A} \approx \mathsf{H}_1$). Let $\delta = \varepsilon' \cdot r \cdot 2^{S-1} = 2^{-\Theta(S)}$. Then

$$\text{density}\{x \in \{0,1\}^n : \mathsf{A}(\mathcal{P}, v_0, x, T) \not\sim_\delta \mathsf{H}_1(\mathcal{P}, v_0, x, T)\} \leq 2^{-S^c}. \tag{15}$$

Proof. Fix any $b \in [B]$ and $v \in V(\mathcal{P})$. Let $I = I_b$, and fix any $x' \in \{0,1\}^n$ with $x'|_I = 0^{|I|}$. Define $f : \{0,1\}^s \to V$ by

$$f(y') = \mathcal{P}|_{[n]\setminus I}(v; x', \mathsf{NisGen}(y')). \tag{16}$$

By Proposition 7,

$$\#\{x|_I \in \{0,1\}^\ell : f(\mathsf{SUExt}(x|_I, U_d)) \not\sim_{\varepsilon' 2^{S-1}} f(U_s)\} \leq 2^{k+S+1}. \tag{17}$$

Therefore,

$$\#\{x \in \{0,1\}^n : x|_{[n]\setminus I} = x'|_{[n]\setminus I} \text{ and } f(\mathsf{SUExt}(x|_I, U_d)) \not\sim_{\varepsilon' 2^{S-1}} f(U_s)\} \leq 2^{k+S+1}. \tag{18}$$

Now, let $M[x]$ be the $\text{size}(\mathcal{P}) \times \text{size}(\mathcal{P})$ stochastic matrix defined by

$$M[x]_{uv} = \Pr_{b,y}[\mathcal{P}|_{[n]\setminus I}(u; x, \mathsf{NisGen}(\mathsf{SUExt}(x|_I, y))) = v \text{ where } I = I_b]. \tag{19}$$

Let $M'[x]$ be the stochastic matrix defined by

$$M'[x]_{uv} = \Pr_{b,y'}[\mathcal{P}|_{[n]\setminus I}(u; x, \mathsf{NisGen}(y')) = v \text{ where } I = I_b]. \tag{20}$$

By summing over all b, v, x', we find that

$$\#\{x \in \{0,1\}^n : M[x] \not\sim_{\varepsilon' 2^{S-1}} M'[x]\} \leq B \cdot 2^S \cdot 2^{n-\ell} \cdot 2^{k+S+1} \tag{21}$$

$$\leq 2^{n - S^{c+1} + O(S^6)} \tag{22}$$

$$\leq 2^{n-S^c}, \tag{23}$$

assuming $c \geq 6$ and n is sufficiently large. If $M[x] \sim_{\varepsilon' 2^{S-1}} M'[x]$, then by Lemma 10, $M[x]^r \sim_\delta M'[x]^r$. The output of A is a sample from $(M[x]^r)_{v_0}$ and the output of H_1 is a sample from $(M'[x]^r)_{v_0}$, completing the proof. ◁

The second hybrid distribution is defined by the algorithm H_2 given by Algorithm 3.

Algorithm 3: The algorithm H_2 defining the second hybrid distribution used to prove Equation (9). The only difference between H_1 and H_2 is that H_2 feeds true randomness to $\mathcal{P}|_{[n]\setminus I}$, instead of feeding it a pseudorandom string from Nisan's generator.

Initialize $v \leftarrow v_0$
repeat r **times**
  Pick $b \in [B]$ uniformly at random and let $I \leftarrow I_b$
  Pick $y'' \in \{0,1\}^T$ uniformly at random
  Update $v \leftarrow \mathcal{P}|_{[n]\setminus I}(v; x, y'')$
end
return v

Algorithm 4: The algorithm H_3 defining the third hybrid distribution used to prove Equation (9). The only difference between H_2 and H_3 is that H_2 terminates after r iterations, whereas H_3 waits until it reaches a terminal vertex of $\mathcal{P}$.

Initialize $v \leftarrow v_0$
while v *is not a terminal vertex of* $\mathcal{P}$ **do**
  Pick $b \in [B]$ uniformly at random and let $I \leftarrow I_b$
  Pick $y'' \in \{0,1\}^T$ uniformly at random
  Update $v \leftarrow \mathcal{P}|_{[n]\setminus I}(v; x, y'')$
end
return v

▷ Claim 12 ($\mathsf{H}_1 \approx \mathsf{H}_2$). For every x,

$$\mathsf{H}_1(\mathcal{P}, v_0, x, T) \sim_{\varepsilon r} \mathsf{H}_2(\mathcal{P}, v_0, x, T). \tag{24}$$

Proof. This follows immediately from the correctness of NisGen and an application of Lemma 10 that is perfectly analogous to the reasoning used to prove Claim 11. ◁

Next, we must show that the output of H_2 is statistically close to the output of H_3. The idea is that in each iteration, with high probability, H_2 progresses by roughly B steps before running into a vertex v with $i(v) \in I$. (Recall that $i(v)$ is the index of the input queried by vertex v.) Therefore, in total, with high probability, H_2 progresses roughly rB steps, which is at least T by our choice of r. We now give the detailed statement and proof.

▷ Claim 13 ($\mathsf{H}_2 \approx \mathsf{H}_3$). For every x,

$$\mathsf{H}_2(\mathcal{P}, v_0, x, T) \sim_{\exp(-r/8)} \mathsf{H}_3(\mathcal{P}, v_0, x, T). \tag{25}$$

Proof. Consider iteration t of the loop in H_2, where $1 \le t \le r$. Let T_t be the number of steps through $\mathcal{P}|_{[n]\setminus I}$ that are taken in iteration t when updating $v = \mathcal{P}|_{[n]\setminus I}(v; x, y'')$ before reaching a vertex that tries to query from I. (If we never reach such a vertex, i.e., we reach a terminal vertex of $\mathcal{P}$, then let $T_t = T$.) We claim that

$$\Pr\left[\sum_{t=1}^{r} T_t < T\right] \le e^{-r/8}. \tag{26}$$

Proof: For $t \in [r]$, consider the value of v at the beginning of iteration t and the string $y'' \in \{0,1\}^T$ chosen in iteration t. As a thought experiment, consider computing $\mathcal{P}(v; x, y'')$,

i.e., taking a walk through the *unrestricted* program. Let $v = u_0, u_1, u_2, \ldots, u_{T'}$ be the vertices visited in this walk, $T' \leq T$. Let S_t be the set of blocks $b' \in [B]$ that are queried by the first $B/2$ steps of this walk. That is,

$$S_t = \{b' \in [B] : \exists h < \lfloor B/2 \rfloor \text{ such that } i(u_h) \in I_{b'}\}, \tag{27}$$

so that $|S_t| \leq \lfloor B/2 \rfloor$. Let $S'_t = S_t \cup [B']$, where B' is chosen so that $|S'_t| = \lfloor B/2 \rfloor$. Let E_t be the event that $b \in S'_t$, where b is the value chosen by H_2 in iteration t of the loop.

Since b and y'' are chosen *independently* at random, the events E_t are independent, and $\Pr[E_t] = \frac{\lfloor B/2 \rfloor}{B} \leq 1/2$. Therefore, by Hoeffding's inequality,

$$\Pr[\#E_t \text{ that occur} > (3/4)r] \leq e^{-r/8}. \tag{28}$$

Now, suppose that E_t does not occur. Then $b \notin S'_t$, so $b \notin S_t$. This implies that when updating $v = \mathcal{P}|_{[n]\setminus I}(v; x, y'')$ (taking a walk through the *restricted* program), we either reach a terminal vertex of $\mathcal{P}$ or we take at least $\lfloor B/2 \rfloor$ steps before reaching a vertex that tries to query I. Therefore, $T_t \geq \min\{\lfloor B/2 \rfloor, T\}$. By Equation (11),

$$\frac{r}{4} \cdot \left\lfloor \frac{B}{2} \right\rfloor \geq r \cdot \left(\frac{B}{8} - 1 \right) \geq T. \tag{29}$$

Equation (26) follows. Since $T \geq \text{length}(\mathcal{P})$, $\sum_{t=1}^{r} T_t \geq T$ implies that H_2 outputs a terminal vertex of $\mathcal{P}$. Therefore, any random string that gives $\sum_{t=1}^{r} T_t \geq T$ also causes H_2 and H_3 to output the same vertex. ◁

Finally, we argue that H_3 perfectly simulates $\mathcal{P}$ (with zero error).

▷ Claim 14 ($\mathsf{H}_3 \sim \mathcal{P}$). For every x,

$$\mathsf{H}_3(\mathcal{P}, v_0, x, T) \sim \mathcal{P}(v_0; x, U_T). \tag{30}$$

Proof. For any path $v_0, v_1, \ldots, v_{T'}$ through $\mathcal{P}$ ending at a terminal vertex, both computations, $\mathsf{H}_3(\mathcal{P}, v_0, x, T)$ and $\mathcal{P}(v_0; x, U_T)$, have exactly a $2^{-T'}$ chance of following that path. ◁

Proof of Theorem 8. By Claims 11 to 14 and the triangle inequality,

$$\text{density}\{x \in \{0,1\}^n : \mathsf{A}(\mathcal{P}, v_0, x, T) \not\sim_\delta \mathcal{P}(v_0; x, U_T)\} \leq 2^{-S^c}, \tag{31}$$

where $\delta = \varepsilon r + \varepsilon' r \cdot 2^{S-1} + e^{-r/8}$. By our choice of ε (Equation (12)), the first term is at most $e^{-cS}/4$. By our choice of ε' (Equation (13)), the second term is also at most $e^{-cS}/4$. By our choice of r (Equation (11)), the third term is at most $e^{-cS}/2$. Therefore, $\delta \leq e^{-cS}$. ◀

3.2 Main Result: Derandomizing Uniform Random-Access Algorithms

Theorem 8 immediately implies $\mathbf{BPTISP}(n \cdot \text{poly}(S), S)$ can be simulated by a typically-correct algorithm that runs in time $n \cdot \text{poly}(S)$ and space $O(S)$ that uses only $\text{poly}(S)$ random bits.

▶ **Corollary 15.** *Fix a function $S(n) \geq \log n$ that is constructible in time $n \cdot \text{poly}(S)$ and space $O(S)$, and fix a constant $c \in \mathbb{N}$. For every language $L \in \mathbf{BPTISP}(n \cdot \text{poly}(S), S)$, there is a randomized algorithm $\mathcal{A}$ running in time $n \cdot \text{poly}(S)$ and space $O(S)$ that uses $\text{poly}(S)$ random bits such that*

$$\text{density}\{x \in \{0,1\}^n : \Pr[\mathcal{A}(x) \neq L(x)] > 2^{-S^c}\} \leq 2^{-S^c}. \tag{32}$$

Proof. Let $\mathcal{B}$ be the algorithm witnessing $L \in \mathbf{BPTISP}(n \cdot \mathrm{poly}(S), S)$. Let c' be a constant so that $\mathcal{B}$ runs in time $n \cdot S^{c'}$. For $n \in \mathbb{N}$, let $\mathcal{P}_n$ be a randomized branching program, where each vertex in $V(\mathcal{P}_n)$ describes a configuration of $\mathcal{B}$ with at most $S(n)$ symbols written on each tape. For each vertex $v \in V(\mathcal{P}_n)$, let $i(v)$ be the location of the input tape read head in the configuration described by v, and let $j(v)$ be the location of the random tape read head in the configuration described by v. The transitions of $\mathcal{P}_n$ correspond to the transitions of $\mathcal{B}$ in the obvious way.

By construction, $\mathcal{P}_n$ is an R-OW branching program with size $2^{O(S)}$ and length at most $n \cdot S^{c'}$. Furthermore, given a vertex v, the neighborhood of v can be computed in $\mathrm{poly}(S)$ time and $O(S)$ space, simply by consulting the transition function for $\mathcal{B}$.

Given $x \in \{0,1\}^n$, the algorithm $\mathcal{A}_0$ runs the algorithm of Theorem 8 on input $(\mathcal{P}_n, v_0, x, n \cdot S^{c'})$, where v_0 encodes the starting configuration of $\mathcal{B}$. This gives a vertex $v \in V(\mathcal{P}_n)$. The algorithm $\mathcal{A}_0$ accepts if and only if v encodes an accepting configuration of $\mathcal{B}$. That way,

$$\mathrm{density}\{x \in \{0,1\}^n : \Pr[\mathcal{A}_0(x) \neq L(x)] > 1/3 + e^{-cS}\} \leq 2^{-S^c}. \tag{33}$$

The algorithm $\mathcal{A}(x)$ runs $O(S^c)$ repetitions of $\mathcal{A}_0(x)$ and takes a majority vote, driving the failure probability down to 2^{-S^c}.

Clearly, $\mathcal{A}$ runs in time $n \cdot S^{c'} \cdot \mathrm{poly}(S) \cdot S^c = n \cdot \mathrm{poly}(S)$ and space $O(S)$. The number of random bits used by $\mathcal{A}$ is $O(\frac{n \cdot S^{c'}}{n} \cdot \mathrm{poly}(S) \cdot S^c) = \mathrm{poly}(S)$. ◀

We can further reduce the randomness complexity by using a pseudorandom generator by Nisan and Zuckerman [30].

▶ **Theorem 16** ([30]). *Fix constants $c \in \mathbb{N}, \alpha > 0$. For every $S \in \mathbb{N}$, there is a generator $\mathsf{NZGen} : \{0,1\}^s \to \{0,1\}^{S^c}$ with seed length $s \leq O(S)$ such that if $\mathcal{P}$ is an R-OW randomized branching program of size 2^S, v is a vertex, and x is an input, then*

$$\mathcal{P}(v; x, \mathsf{NZGen}(U_s)) \sim_\varepsilon \mathcal{P}(v; x, U_{S^c}), \tag{34}$$

where $\varepsilon = 2^{-S^{1-\alpha}}$. Given S and z, $\mathsf{NZGen}(z)$ can be computed in $O(S)$ space and $\mathrm{poly}(S)$ time.

▶ **Corollary 17** (Main result). *Fix a function $S(n) \geq \log n$ that is constructible in time $n \cdot \mathrm{poly}(S)$ and space $O(S)$, and fix constants $c \in \mathbb{N}, \alpha > 0$. For every language $L \in \mathbf{BPTISP}(n \cdot \mathrm{poly}(S), S)$, there is a randomized algorithm $\mathcal{A}$ running in time $n \cdot \mathrm{poly}(S)$ and space $O(S)$ that uses $O(S)$ random bits such that*

$$\mathrm{density}\{x \in \{0,1\}^n : \Pr[\mathcal{A}(x) \neq L(x)] > 2^{-S^{1-\alpha}}\} \leq 2^{-S^c}. \tag{35}$$

Proof sketch. Compose the algorithm of Corollary 15 with the Nisan-Zuckerman generator (Theorem 16). The algorithm of Corollary 15 can be implemented as a randomized branching program as in the proof of Corollary 15. ◀

Finally, we can eliminate the random bits entirely at the expense of time.

▶ **Corollary 18.** *For every space-constructible function $S(n) \geq \log n$, for every constant $c \in \mathbb{N}$,*

$$\mathbf{BPTISP}(n \cdot \mathrm{poly}(S), S) \text{ is within } 2^{-S^c} \text{ of } \mathbf{DSPACE}(S). \tag{36}$$

Proof. Run the algorithm of Corollary 17 on all possible random strings and take a majority vote. ◀

4 Derandomizing Turing Machines

In this section, we give our improved typically-correct derandomizations for Turing machines. Sections 4.1 and 4.2 concern derandomization with a low mistake rate, and Sections 4.3 to 4.5 concern derandomization of Turing machines with runtime $T \approx n^2$.

4.1 Low-Randomness Simulation of Sequential-Access Branching Programs with a Low Mistake Rate

Recall that for a nonterminal vertex v in a branching program, $i(v)$ is the index of the input queried by v, and $j(v)$ is the index of the random string queried by v.

▶ **Definition 19.** *An* S-OW randomized branching program *is a randomized branching program $\mathcal{P}$ such that for every edge (v, v') between two nonterminal vertices, $|i(v) - i(v')| \leq 1$ and $j(v') \in \{j(v), j(v) + 1\}$.*

In words, an S-OW randomized branching program has *sequential* access to its input and one-way access to its random bits. By "sequential access", we mean that after reading bit i, it reads bit $i - 1$, bit i, or bit $i + 1$, like a head of a Turing machine. For S-OW branching programs, we give an algorithm analogous to Theorem 8 but with a much lower rate of mistakes.

▶ **Theorem 20.** *For each constant $c \in \mathbb{N}$, there is a randomized random-access algorithm* A *with the following properties. Suppose $\mathcal{P}$ is an S-OW randomized branching program on $\{0,1\}^n \times \{0,1\}^T$ with $S \geq \log n$, where $S \stackrel{\text{def}}{=} \lceil \log \text{size}(\mathcal{P}) \rceil$. Suppose $v_0 \in V(\mathcal{P})$, $T \geq \text{length}(\mathcal{P})$, and $x \in \{0,1\}^n$. Then $\mathsf{A}(\mathcal{P}, v_0, x, T)$ outputs a vertex $v \in V(\mathcal{P})$. The number of random bits used by A is $\lceil T/n \rceil \cdot \text{poly}(S)$, and A runs in time*[7] *$T \cdot \text{poly}(n, S)$ and space $O(S)$. Finally, for every such $\mathcal{P}, v_0, T$,*

$$\#\{x \in \{0,1\}^n : \mathsf{A}(\mathcal{P}, v_0, x, T) \not\approx_{\exp(-cS)} \mathcal{P}(v_0; x, U_T)\} \leq 2^{n/S^c}. \tag{37}$$

The proof of Theorem 20 is very similar to the proof of Theorem 8. The main difference is that instead of using a *small* part of the input as the source of randomness, we use *most* of the input as a source of randomness. The only part of the input that is *not* used as a source of randomness is the region near the bit that the branching program was processing at the beginning of the current phase.

Because the proof of Theorem 20 does not introduce any significantly new techniques, we defer the proof to Appendix D.

4.2 Derandomizing Turing Machines with a Low Mistake Rate

A *randomized Turing machine* is defined like a randomized random-access Turing machine except that there are no index tapes. Thus, moving a read head from position i to position j takes $|i - j|$ steps. For functions $T, S : \mathbb{N} \to \mathbb{N}$, let $\mathbf{BPTISP}_{\text{TM}}(T, S)$ denote the class of languages L such that there is a randomized Turing machine $\mathcal{A}$ that always runs in time $O(T(n))$ and space $O(S(n))$ such that for every $x \in \{0,1\}^*$,

$$\Pr[\mathcal{A}(x) = L(x)] \geq 2/3. \tag{38}$$

[7] Like in Theorem 8, the graph of $\mathcal{P}$ should be encoded in adjacency list format. We also stress that A is a random-access simulation of sequential-access branching programs.

Trivially, a randomized Turing machine can be simulated by a randomized random-access Turing machine without loss in efficiency. Conversely, a single step of a randomized $O(S)$-space random-access Turing machine can be simulated in $O(n+S)$ steps by a randomized Turing machine. This proves the following elementary containments.

▶ **Proposition 21.** *For any functions $T, S : \mathbb{N} \to \mathbb{N}$ with $S(n) \geq \log n$,*

$$\mathbf{BPTISP}_{\mathrm{TM}}(T,S) \subseteq \mathbf{BPTISP}(T,S) \subseteq \mathbf{BPTISP}_{\mathrm{TM}}(T \cdot (n+S), S). \tag{39}$$

Theorem 20 combined with the Nisan-Zuckerman generator [30] immediately implies a derandomization theorem for Turing machines analogous to Corollary 17.

▶ **Corollary 22.** *Fix a function $S : \mathbb{N} \to \mathbb{N}$ with $S(n) \geq \log n$ that is constructible in time $\mathrm{poly}(n, S)$ and space $O(S)$, and fix constants $c \in \mathbb{N}, \alpha > 0$. For every language $L \in \mathbf{BPTISP}_{\mathrm{TM}}(n \cdot \mathrm{poly}(S), S)$, there is a randomized algorithm $\mathcal{A}$ running in time $\mathrm{poly}(n, S)$ and space $O(S)$ that uses $O(S)$ random bits such that*

$$\#\{x \in \{0,1\}^n : \Pr[\mathcal{A}(x) \neq L(x)] > 2^{-S^{1-\alpha}}\} \leq 2^{n/S^c}. \tag{40}$$

Proof sketch. A randomized Turing machine obviously gives rise to an S-OW randomized branching program. Like in the proof of Corollary 15 (but with Theorem 20 in place of Theorem 8), we first obtain an algorithm that uses $\mathrm{poly}(S)$ random bits. Composing with the Nisan-Zuckerman generator (Theorem 16) completes the proof. ◀

▶ **Corollary 23.** *For every space-constructible function $S(n) \geq \log n$, for every constant $c \in \mathbb{N}$,*

$$\mathbf{BPTISP}_{\mathrm{TM}}(n \cdot \mathrm{poly}(S), S) \text{ is within } 2^{-n+n/S^c} \text{ of } \mathbf{DSPACE}(S). \tag{41}$$

Proof. Simulate the algorithm of Corollary 22 on all possible random strings and take a majority vote. ◀

4.3 Simulating Branching Programs with Random Access to Random Bits

We now move on to our second derandomization of Turing machines, as outlined in Section 1.2.4. Recall that for a nonterminal vertex v in a branching program, $i(v)$ is the index of the input that is queried by v.

▶ **Definition 24.** *An* S-R *randomized branching program is a randomized branching program $\mathcal{P}$ such that for every edge (v, v') between two nonterminal vertices, $|i(v) - i(v')| \leq 1$.*

In words, an S-R randomized branching program has sequential access to its input and random access to its random bits. This model is *more general* than the S-OW model; the S-OW model corresponds more directly to the randomized Turing machine model. But studying the more general S-R model will help us derandomize Turing machines.

We will give a randomness-efficient algorithm for simulating S-R randomized branching programs, roughly analogous to Theorems 8 and 20. The simulation will only work well if the branching program has small length *and* uses few random bits.

Our simulation of S-R randomized branching programs is a fairly straightforward application of work by Kinne et al. [24]; this section is not technically novel. But it is useful to be able to compare the work by Kinne et al. [24] to our algorithms based on the "out of sight, out of mind" technique.

Unlike Theorems 8 and 20, our simulation of S-R branching programs will not work on a step-by-step basis, generating a distribution on vertices that approximates the behavior of the branching program. Instead, our simulation of S-R branching programs will only work for S-R branching programs that *compute a Boolean function*. We now give the relevant definition.

▶ **Definition 25.** *Let $\mathcal{P}$ be a randomized branching program on $\{0,1\}^n \times \{0,1\}^m$. Suppose some vertex $v_0 \in V(\mathcal{P})$ is labeled as the* start vertex*, and every terminal vertex of $\mathcal{P}$ is labeled with an* output bit *$b \in \{0,1\}$. In this case, we identify $\mathcal{P}$ with a function $\mathcal{P} : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}$ defined by*

$$\mathcal{P}(x,y) = \text{the output bit labeling } \mathcal{P}(v_0; x, y). \tag{42}$$

We say that $\mathcal{P}$ computes *$f : \{0,1\}^n \to \{0,1\}$ with failure probability δ if for every $x \in \{0,1\}^n$,*

$$\Pr[\mathcal{P}(x, U_m) = f(x)] \geq 1 - \delta. \tag{43}$$

Instead of assuming a time bound, it will be useful to assume a bound on the *query complexity* of the branching program.

▶ **Definition 26.** *Let $\mathcal{P}$ be randomized branching program. The* query complexity *of $\mathcal{P}$, denoted* $\text{queries}(\mathcal{P})$*, is the maximum, over all paths $v_1, v_2, \ldots, v_T$ through $\mathcal{P}$ consisting entirely of nonterminal vertices, of*

$$1 + \#\{t \in \{2, 3, \ldots, T\} : i(v_t) \neq i(v_{t-1})\}. \tag{44}$$

In words, $\text{queries}(\mathcal{P})$ is the number of steps that $\mathcal{P}$ takes in which it queries a *new* bit of its input, i.e., not the bit that it queried in the previous step. Trivially, $\text{queries}(\mathcal{P}) \leq \text{length}(\mathcal{P})$. The reader is encouraged to think of the distinction between $\text{queries}(\mathcal{P})$ and $\text{length}(\mathcal{P})$ as being a technicality that can be ignored on the first reading.

We can now state our deterministic simulation theorem for S-R randomized branching programs. It consists of a method of deterministically generating coins for the branching program from its input.

▶ **Theorem 27.** *There is a constant $\alpha > 0$ so that for every n, m with $m \leq n/3$, there is a function $\mathsf{R} : \{0,1\}^n \to \{0,1\}^m$ with the following properties. Suppose $\mathcal{P}$ is an S-R randomized branching program on $\{0,1\}^n \times \{0,1\}^m$ that computes a function f with failure probability δ. Suppose $TSm \leq \alpha n^2$, where $T \stackrel{\text{def}}{=} \text{queries}(\mathcal{P})$ and $S \stackrel{\text{def}}{=} \lceil \log \text{size}(\mathcal{P}) \rceil$. Then*

$$\text{density}\{x \in \{0,1\}^n : \mathcal{P}(x, \mathsf{R}(x)) \neq f(x)\} \leq 3\delta + m \cdot 2^{-\alpha n/m}. \tag{45}$$

Furthermore, given x and m, $\mathsf{R}(x)$ can be computed in space $O(\log n)$.

The function R is based on a pseudorandom generator by Kinne et al. [24] for *multiparty communication protocols*. In a *public-coin randomized 3-party NOF protocol* Π, there are three parties, three inputs x_1, x_2, x_3, and one random string y. Party i knows x_j for $j \neq i$, and all three parties know y. All parties have access to a blackboard. The protocol specifies who should write next as a function of what has been written on the blackboard so far and y. Eventually, the protocol specifies the output $\Pi(x_1, x_2, x_3, y)$, which should be a function of what has been written on the blackboard and y. The communication complexity of Π is the maximum number of bits written on the blackboard over all x_1, x_2, x_3, y. A *deterministic 3-party NOF protocol* is just the case $|y| = 0$.

Following Kinne et al. [24], we rely on a 3-party communication complexity lower bound by Babai et al. [7]. For an integer $\ell \in \mathbb{N}$, define $\mathrm{GIP}_\ell : (\{0,1\}^\ell)^3 \to \{0,1\}$ to be the generalized inner product function, i.e.,

$$\mathrm{GIP}_\ell(x,y,z) = \sum_{i=1}^{\ell} x_i y_i z_i \bmod 2. \tag{46}$$

Babai et al. showed that the trivial communication protocol for GIP_ℓ is essentially optimal, even in the average-case setting.

▶ **Theorem 28** ([7]). *There is a constant $\beta > 0$ so that for every $\ell \in \mathbb{N}, \varepsilon > 0$, if Π is a deterministic 3-party NOF protocol with*

$$\Pr_{x,y,z}[\Pi(x,y,z) = \mathrm{GIP}_\ell(x,y,z)] \geq \frac{1}{2} + \varepsilon, \tag{47}$$

then the communication complexity of Π is at least $\beta \cdot (\ell - \log(1/\varepsilon))$.

To define R, let $x \in \{0,1\}^n$. Partition $n = n_1 + n_2 + n_3$, where $n_i \geq \lfloor n/3 \rfloor$ for each i. Correspondingly partition $x = x_1 \circ x_2 \circ x_3$, where $|x_i| = n_i$. Define

$$\ell = \left\lfloor \frac{\lfloor n/3 \rfloor}{m} \right\rfloor, \tag{48}$$

so that $\ell \geq 1$. For $i \in [3]$ and $j \in [m]$, let x_{ij} be the jth ℓ-bit substring of x_i. (Note that due to roundoff errors, for some values of n, some bits of x are not represented in any x_{ij}.) Then we define

$$\mathsf{R}(x) = \mathrm{GIP}_\ell(x_{11}, x_{21}, x_{31}) \circ \cdots \circ \mathrm{GIP}_\ell(x_{1m}, x_{2m}, x_{3m}) \in \{0,1\}^m. \tag{49}$$

Kinne et al. observed that $x \mapsto (x, \mathsf{R}(x))$ is a pseudorandom generator that fools 3-party NOF protocols [24]. For clarity, we reproduce the argument here.

▶ **Lemma 29.** *Suppose $\Pi : \{0,1\}^{n_1} \times \{0,1\}^{n_2} \times \{0,1\}^{n_3} \times \{0,1\}^m \to \{0,1\}$ is a public-coin randomized 3-party NOF protocol. Suppose that for some $\varepsilon > 0$, Π uses less than $\beta \cdot (\ell - \log(1/\varepsilon))$ bits of communication, where β is the constant of Theorem 28. Then*

$$\left| \Pr_{x_1,x_2,x_3,y}[\Pi(x_1,x_2,x_3,y) = 1] - \Pr_{x_1,x_2,x_3}[\Pi(x_1,x_2,x_3,\mathsf{R}(x_1,x_2,x_3)) = 1] \right| < \varepsilon m. \tag{50}$$

Proof. Let

$$\delta = \left| \Pr_{x_1,x_2,x_3,y}[\Pi(x_1,x_2,x_3,y) = 1] - \Pr_{x_1,x_2,x_3}[\Pi(x_1,x_2,x_3,\mathsf{R}(x_1,x_2,x_3)) = 1] \right|. \tag{51}$$

By Yao's distinguisher-to-predictor argument [42], there is some index $i \in [m]$ and a protocol $\Pi' : \{0,1\}^{n_1} \times \{0,1\}^{n_2} \times \{0,1\}^{n_3} \times \{0,1\}^{i-1} \to \{0,1\}$ so that

$$\Pr_{x_1,x_2,x_3}[\Pi'(x_1,x_2,x_3,\mathsf{R}(x_1,x_2,x_3)|_{[i-1]}) = \mathsf{R}(x_1,x_2,x_3)_i] \geq \frac{1}{2} + \frac{\delta}{m}. \tag{52}$$

The protocol Π' is a public-coin randomized 3-party NOF protocol that still uses less than $\beta \cdot (\ell - \log(1/\varepsilon))$ bits of communication, since it merely involves simulating Π with certain input/coin bits fixed to certain values and possibly negating the output. This immediately implies a protocol for GIP_ℓ with the same parameters with advantage δ/m. There is some way to fix the randomness to preserve advantage, so by Theorem 28, $\delta/m < \varepsilon$. ◀

The connection between S-R randomized branching programs and 3-party communication protocols is given by the following lemma.

▶ **Lemma 30.** *There is a public-coin randomized 3-party NOF protocol* $\Pi : \{0,1\}^{n_1} \times \{0,1\}^{n_2} \times \{0,1\}^{n_3} \times \{0,1\}^m \to \{0,1\}$ *such that*

$$\Pi(x_1, x_2, x_3, y) = \mathcal{P}(x_1 \circ x_2 \circ x_3, y), \tag{53}$$

and Π *uses only* $O(\frac{TS}{n})$ *bits of communication.*

Proof. Parties 1 and 3 alternate simulating the operation of $\mathcal{P}$. If party 1 is simulating and the program reads from the first n_1 bits of the input, party 1 sends the state to party 3. Similarly, if party 3 is simulating and the program reads from the last n_3 bits of the input, party 3 sends the state to party 1. Each such transition indicates that the program must have spent at least n_2 steps traversing the middle n_2 bits of the input. Therefore, the total number of such transitions is at most $\frac{T}{n_2}$. ◀

Given Lemmas 29 and 30, Theorem 27 follows by a lemma by Kinne et al. [24, Lemma 1]. For clarity, we reproduce the argument here.

Proof of Theorem 27. The best case is at least as good as the average case, so there is some string $y_* \in \{0,1\}^m$ such that

$$\Pr_{x \in \{0,1\}^n}[\mathcal{P}(x, y_*) \neq f(x)] \leq \delta. \tag{54}$$

Define $g : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}$ by

$$g(x,y) = \begin{cases} 1 & \text{if } \mathcal{P}(x,y) = \mathcal{P}(x,y_*) \\ 0 & \text{otherwise.} \end{cases} \tag{55}$$

Think of $x \in \{0,1\}^n$ as $x = x_1 \circ x_2 \circ x_3$, like in the definition of R. Then by Lemma 30, g can be computed by a 3-party NOF protocol using $O(\frac{TS}{n})$ bits of communication. By choosing α small enough and setting $\varepsilon = 2^{-\alpha n/m}$, this protocol for f will use fewer than $\beta(\ell - \log(1/\varepsilon))$ bits of communication. Therefore, by Lemma 29,

$$\Pr_x[\mathcal{P}(x, \mathsf{R}(x)) \neq \mathcal{P}(x, y_*)] \leq \Pr_{x,y}[\mathcal{P}(x,y) \neq \mathcal{P}(x, y_*)] + \varepsilon m. \tag{56}$$

Therefore,

$$\begin{aligned} \Pr_x[\mathcal{P}(x, \mathsf{R}(x)) \neq f(x)] &\leq \Pr_x[\mathcal{P}(x, y_*) \neq f(x)] + \Pr_x[\mathcal{P}(x, \mathsf{R}(x)) \neq \mathcal{P}(x, y_*)] && (57) \\ &\leq \delta + \Pr_{x,y}[\mathcal{P}(x,y) \neq \mathcal{P}(x, y_*)] + \varepsilon m && (58) \\ &\leq \delta + \Pr_{x,y}[\mathcal{P}(x,y) \neq f(x)] + \Pr_x[\mathcal{P}(x, y_*) \neq f(x)] + \varepsilon m && (59) \\ &\leq \delta + \delta + \delta + \varepsilon m. && (60) \end{aligned}$$

Obviously, $\mathsf{R}(x)$ can be computed in $O(\log n)$ space. ◀

4.4 Randomness-Efficient Amplification for Branching Programs

We will use a space-efficient expander walk algorithm by Gutfreund and Viola [17].

▶ **Theorem 31** ([17]). *For every $s \in \mathbb{N}$, there is a constant-degree expander graph G on vertex set $\{0,1\}^s$. Furthermore, there is an algorithm* GVWalk *such that if $y \in \{0,1\}^s$ is a vertex and $e_1, e_2, \ldots, e_r \in \{0,1\}^{O(1)}$ are edge labels, then* $\mathsf{GVWalk}(y, e_1, e_2, \ldots, e_r)$ *outputs the vertex reached by starting at y and taking a walk by following the edge labels $e_1, e_2, \ldots, e_r$. The algorithm* GVWalk *runs in space $O(\log s + \log r)$.*

Recall that we are working toward derandomizing the class $\mathbf{BPTISP}_{\mathrm{TM}}(T,S)$ for all $TS^2 \le o(n^2/\log n)$. This class corresponds to branching programs on $\{0,1\}^n \times \{0,1\}^T$ that compute some function with failure probability $1/3$. But Theorem 27 requires that the branching program use at most $\frac{\alpha n^2}{TS}$ random bits. Furthermore, the failure probability of the branching program governs the mistake rate of the derandomization.

We can overcome these two difficulties because randomized Turing machines correspond to *S-OW* randomized branching programs (i.e., programs that have sequential access to the input and one-way access to the random bits), whereas Theorem 27 applies to the more powerful S-R model (i.e., programs that have sequential access to the input and *random* access to the random bits). An S-OW branching program can be simulated by an S-R branching program using very few random bits by applying Nisan's generator. The following lemma combines this idea with a random walk on an expander graph (Theorem 31) for amplification. This is the same technique that Fortnow and Klivans used to prove that $\mathbf{BPL} \subseteq \mathbf{L}/O(n)$ [12].

▶ **Lemma 32.** *Suppose $\mathcal{P}$ is an S-OW randomized branching program on $\{0,1\}^n \times \{0,1\}^T$ that computes a function $f : \{0,1\}^n \to \{0,1\}$ with failure probability $1/3$. Let $S = \log \mathrm{size}(\mathcal{P})$. For every $\delta > 0$, there is an S-R branching program $\mathcal{P}'$ on $\{0,1\}^n \times \{0,1\}^m$ that computes f with failure probability δ such that*

$$\mathrm{queries}(\mathcal{P}') \le O((\mathrm{queries}(\mathcal{P}) + n)\log(1/\delta)), \tag{61}$$

$$\log \mathrm{size}(\mathcal{P}') \le O(S + \log\log(1/\delta)), \tag{62}$$

$$m \le O(S \log T + \log(1/\delta)). \tag{63}$$

Furthermore, given $\mathcal{P}$, δ, and a vertex $v \in V(\mathcal{P}')$, the neighborhood of v can be computed in time[8] $\mathrm{poly}(S, \log(1/\delta))$ *and space $O(S + \log\log(1/\delta))$.*

Proof. Let $\mathsf{NisGen} : \{0,1\}^s \to \{0,1\}^T$ be Nisan's generator with error 0.1 for randomized branching programs of size $\mathrm{size}(\mathcal{P})$. Let G be the expander of Theorem 31 on vertex set $\{0,1\}^s$. We will interpret a string $y \in \{0,1\}^m$ as describing a walk through G from an arbitrary initial vertex of length $r-1$, so that $m = s + O(r)$. Let $y_1, \ldots, y_r \in \{0,1\}^s$ be the vertices visited by this walk. The program $\mathcal{P}'(x,y)$ runs $\mathcal{P}(x, \mathsf{NisGen}(y_t))$ for every $y \in [r]$ and takes a majority vote of the answers; it finds the vertices y_t by running the algorithm GVWalk of Theorem 31. By the expander walk Chernoff bound [14], for an appropriate choice of $r = \Theta(\log(1/\delta))$, the failure probability of $\mathcal{P}'$ is at most δ.

Clearly, $\mathrm{queries}(\mathcal{P}') \le r \cdot (\mathrm{queries}(\mathcal{P}) + n)$, where the $+n$ term takes care of the steps needed to get from the final position of x read in one iteration of $\mathcal{P}$ to the first position of x read in the next iteration of $\mathcal{P}$ (recall that $\mathcal{P}'$ is an S-R branching program).

[8] As usual, we assume that the graph of $\mathcal{P}$ is encoded in adjacency list format. We also assume that the start vertex v_0 is designated in a way that allows it to be computed in the specified time and space.

The space needed by $\mathcal{P}'$ consists of the S bits of space needed for $\mathcal{P}$, plus $O(S)$ bits of space for computing NisGen, plus $O(\log r)$ bits of space to keep track of the answers generated by the iterations, plus $O(\log S + \log r)$ bits of space for GVWalk. Finally, computing the neighborhood of v merely requires inspecting the transition functions for the algorithms NisGen and GVWalk, inspecting $\mathcal{P}$, and doing arithmetic. ◀

4.5 Derandomizing Turing Machines with Runtime Near n^2

Finally, we are ready to state and prove our typically-correct derandomization of Turing machines based on Theorem 27.

▶ **Corollary 33.** *Suppose $T, S : \mathbb{N} \to \mathbb{N}$ are both constructible in time $\mathrm{poly}(n)$ and space $O(S)$ and $TS^2 \leq o\left(\frac{n^2}{\log n}\right)$. For every language $L \in \mathbf{BPTISP}_{\mathrm{TM}}(T, S)$, there is a constant $\gamma > 0$ so that*

$$L \text{ is within } \exp\left(-\frac{\gamma n}{\sqrt{TS}}\right) + \exp\left(-\frac{\gamma n^2}{TS^2 \log n}\right) \text{ of } \mathbf{DTISP}(\mathrm{poly}(n), S). \tag{64}$$

The rate of mistakes in Corollary 33 is always $o(1)$. The rate of mistakes gets smaller (i.e., the simulation quality gets higher) when T and S are smaller. For example, if $S = \log n$ and $T = n^2/\log^4 n$, the rate of mistakes in Equation (64) is $n^{-\Omega(1)}$. For another example, if $S = \mathrm{polylog}\, n$ and $T = n\,\mathrm{polylog}\, n$, the rate of mistakes in Equation (64) is $\exp\left(-\widetilde{\Omega}(\sqrt{n})\right)$.

As a reminder, Corollary 33 is incomparable to Corollary 18: the randomized classes in the two results are incomparable; the deterministic algorithm in Corollary 33 is faster; the mistake rate in Corollary 33 is lower when S and T are not too big. Similarly, Corollary 33 is incomparable to Corollary 23: the randomized class in Corollary 33 is more powerful and the deterministic algorithm in Corollary 33 is faster, but the mistake rate in Corollary 33 is much higher. Finally, even when $S \geq n^{\Omega(1)}$, Corollary 33 is incomparable to derandomizing via the Nisan-Zuckerman generator [30], because the deterministic algorithm of Corollary 33 runs in polynomial time, although it makes some mistakes.

Conceptually, the proof of Corollary 33 merely consists of combining Lemma 32 and Theorem 27. The only work to be done is in appropriately choosing δ and verifying parameters.

Proof of Corollary 33. Let $\mathcal{A}$ be the algorithm witnessing $L \in \mathbf{BPTISP}_{\mathrm{TM}}(T, S)$. Let $\mathcal{P}_n$ be the S-OW branching program on $\{0,1\}^n \times \{0,1\}^T$ describing the behavior of $\mathcal{A}$ on inputs of length n.

We consider two cases. First, suppose $TS^3 > n^2/\log^2 n$. Then let

$$\delta = \exp\left(-\frac{\gamma_0 n^2}{TS^2 \log n}\right), \tag{65}$$

where the constant γ_0 will be specified later. Let $\mathcal{P}'_n$ be the S-R branching program on $\{0,1\}^n \times \{0,1\}^m$ given by Lemma 32. There is a constant c that does not depend on γ so that

$$\begin{aligned} \mathrm{queries}(\mathcal{P}'_n) \cdot \log \mathrm{size}(\mathcal{P}'_n) \cdot m &\leq cTS^2 \log n \ln(1/\delta) + cTS \ln^2(1/\delta) && (66)\\ &= c\gamma_0 n^2 + \frac{c\gamma_0^2 n^4}{TS^3 \log^2 n} && (67)\\ &\leq c\gamma_0 n^2 + c\gamma_0^2 n^2. && (68) \end{aligned}$$

Choose γ_0 so that $c\gamma_0 + c\gamma_0^2 \leq \alpha$, where α is the value in Theorem 27. Since $TS^2 \leq o(n^2/\log n)$ and $T \geq n$, we must have $S \leq o(\sqrt{n/\log n})$. Therefore,

$$m \leq O\left(S \log n + \frac{n^2}{TS^2 \log n}\right) \leq O\left(S \log n + \frac{TS^3 \log n}{TS^2}\right) \leq o(\sqrt{n \log n}) \leq n/3. \tag{69}$$

Therefore, the hypotheses of Theorem 27 are satisfied.

The deterministic algorithm, naturally, outputs $\mathcal{P}'_n(x, \mathsf{R}(x))$, where R is the function of Theorem 27. It is immediate that this runs in $\text{poly}(n)$ time and $O(S)$ space. Finally, to compute the rate of mistakes, observe that

$$m \cdot 2^{-\alpha n/m} \leq \exp\left(-\Omega\left(-\frac{n}{S \log n}\right)\right), \tag{70}$$

whereas

$$\delta \geq \exp\left(-O\left(\frac{n}{S^2 \log n}\right)\right). \tag{71}$$

Therefore, when n is sufficiently large, $m \cdot 2^{-\alpha n/m} < \delta$. Therefore,

$$\text{density}\{x \in \{0,1\}^n : \mathcal{P}'_n(x, \mathsf{R}(x)) \neq L(x)\} \leq 4\delta. \tag{72}$$

For the second case, suppose $TS^3 \leq n^2/\log^2 n$. Then let

$$\delta = \exp\left(-\frac{\gamma_0 n}{\sqrt{TS}}\right). \tag{73}$$

Again, let $\mathcal{P}'_n$ be the S-R branching program on $\{0,1\}^n \times \{0,1\}^m$ given by Lemma 32. Then

$$\begin{aligned}
\text{queries}(\mathcal{P}'_n) \cdot \log \text{size}(\mathcal{P}'_n) \cdot m &\leq cTS^2 \log n \ln(1/\delta) + cTS \ln^2(1/\delta) && (74)\\
&= c\gamma_1 \sqrt{TS^3} n \log n + c\gamma_1^2 n^2 && (75)\\
&\leq c\gamma_1 n^2 + c\gamma_1^2 n^2 && (76)\\
&\leq \alpha n^2. && (77)
\end{aligned}$$

Furthermore, since $TS^3 \leq n^2/\log^2 n$, taking a square root gives $S\sqrt{TS} \leq n/\log n$, and hence

$$m \leq O\left(S \log n + \frac{n}{\sqrt{TS}}\right) \leq O\left(\frac{n}{\sqrt{TS}}\right) < n/3. \tag{78}$$

Therefore, again, the hypotheses of Theorem 27. In this case as well, the deterministic algorithm outputs $\mathcal{P}'_n(x, \mathsf{R}(x))$. We now compute the rate of mistakes again. We have

$$m \cdot 2^{-\alpha n/m} \leq \exp(-\Omega(\sqrt{TS})) < \delta \tag{79}$$

for sufficiently large n, because $\sqrt{TS} \geq \sqrt{n \log n}$. Therefore, once again,

$$\text{density}\{x \in \{0,1\}^n : \mathcal{P}'_n(x, \mathsf{R}(x)) \neq L(x)\} \leq 4\delta. \tag{80}$$

Choosing $\gamma < \gamma_0$ completes the proof. ◀

5 Derandomization with Advice

As previously mentioned, Fortnow and Klivans showed that $\mathbf{BPL} \subseteq \mathbf{L}/O(n)$ [12]. We now explain how to refine their ideas and slightly improve their result. Fortnow and Klivans' argument relied on the Gutfreund-Viola space-efficient expander walk (Theorem 31). They only used this expander for its sampling properties. Extractors also have good sampling properties. Our improvement will come from simply replacing the expander-based sampler in Fortnow and Klivans' argument with the GUV-based extractor of Theorem 6.

▶ **Theorem 34.** $\mathbf{BPL} \subseteq \mathbf{L}/(n + O(\log^2 n))$.

Proof. Let $\mathcal{A}$ be an algorithm witnessing $L \in \mathbf{BPL}$, and assume $\mathcal{A}$ has failure probability at most 0.1. Let $\mathsf{NisGen} : \{0,1\}^s \to \{0,1\}^{\mathrm{poly}(n)}$ be Nisan's generator (Theorem 9) with error 0.1 and space bound sufficient to fool $\mathcal{A}$, so that $s \leq O(\log^2 n)$. Let $\mathsf{GUVExt} : \{0,1\}^{n+2s+3} \times \{0,1\}^d \to \{0,1\}^s$ be the $(2s, 0.1)$-extractor of Theorem 6, so that $d \leq O(\log n)$.

Given input $x \in \{0,1\}^n$ and advice $a \in \{0,1\}^{n+2s+3}$, run $\mathcal{A}(x, \mathsf{NisGen}(\mathsf{GUVExt}(a, z)))$ for all z and take a majority vote.

This algorithm clearly runs in space $O(\log n)$. By Proposition 7, for each fixed x, the number of advice strings a causing the algorithm to give the wrong answer is at most 2^{2s+2}. Therefore, the total number of advice strings a that cause the algorithm to give the wrong answer for any x is at most $2^{n+2s+2} < 2^{|a|}$. Therefore, there is some choice of a such that the algorithm succeeds on all inputs. ◀

We now generalize Theorem 34, showing that the amount of advice can be reduced to below n in certain cases. We will rely on a special feature of Nisan's generator that Nisan used to prove $\mathbf{RL} \subseteq \mathbf{SC}$. The seed to Nisan's generator is naturally divided into two parts, $s = s_1 + s_2$, where $s_2 \leq O(S + \log(1/\varepsilon))$.[9] Nisan showed that there is an efficient procedure to *check* that the first part of the seed is "good" for a particular randomized log-space algorithm and a particular input to that algorithm.

▶ **Lemma 35** ([28]). *For every $S \in \mathbb{N}$, there is a function $\mathsf{NisGen} : \{0,1\}^{s_1} \times \{0,1\}^{s_2} \to \{0,1\}^{2^S}$, with $s_1 \leq O(S^2)$ and $s_2 \leq O(S)$, and an algorithm Check, so that*

- *For any R-OW randomized branching program $\mathcal{P}$ with $\log \mathrm{size}(\mathcal{P}) \leq S$ and any input $x \in \{0,1\}^n$,*

$$\Pr_{y_1 \in \{0,1\}^{s_1}}[\mathsf{Check}(\mathcal{P}, x, y_1) = 1] \geq 1/2. \tag{81}$$

- *If $\mathsf{Check}(\mathcal{P}, x, y_1) = 1$, then for any vertex $v_0 \in V(\mathcal{P})$,*

$$\mathcal{P}(v_0; x, \mathsf{NisGen}(y_1, U_{s_2})) \sim_{0.1} \mathcal{P}(v_0; x, U_{2^S}). \tag{82}$$

Furthermore, Check runs in space $O(S)$, and given S, y_1, and y_2, $\mathsf{NisGen}(y_1, y_2)$ can be computed in space $O(S)$.

A $\mathbf{ZP} \cdot \mathbf{SPACE}(S)$ *algorithm* for a language L with failure probability δ is a randomized Turing machine $\mathcal{A}$ with *two-way* access to its random bits such that $\mathcal{A}$ runs in space $O(S)$, $\Pr[\mathcal{A}(x) \in \{L(x), \perp\}] = 1$, and $\Pr[\mathcal{A}(x) = \perp] \leq \delta$. The following lemma refines a theorem by Nisan that says that $\mathbf{BPL} \subseteq \mathbf{ZP} \cdot \mathbf{L}$ [27]; the improvement is that our algorithm has a low failure probability relative to the number of random bits it uses.

[9] The first s_1 bits specify the hash functions, and the last s_2 bits specify the input to those hash functions.

▶ **Lemma 36.** *Fix $S : \mathbb{N} \to \mathbb{N}$ with $S(n) \geq \log n$ and $\delta : \mathbb{N} \to [0,1]$, both constructible in space $O(S)$. For every $L \in \mathbf{BPSPACE}(S)$, there is a $\mathbf{ZP} \cdot \mathbf{SPACE}(S)$ algorithm $\mathcal{A}$ that decides L with failure probability δ and uses $\log_2(1/\delta) + O(S^2)$ random bits.*

Proof. Let $\mathcal{B}$ be the algorithm witnessing $L \in \mathbf{BPSPACE}(S)$, and assume $\mathcal{B}$ has failure probability at most 0.1. Let $\mathcal{P}$ be the corresponding R-OW branching program for inputs of length n. Let $\mathsf{NisGen} : \{0,1\}^{s_1} \times \{0,1\}^{s_2} \to \{0,1\}^{\mathrm{poly}(n)}$ be the generator of Lemma 35 with space bound $\lceil \log \mathrm{size}(\mathcal{P}) \rceil$, so that $s_1 \leq O(S^2)$.

Let $\ell = \lceil \log_2(1/\delta) \rceil + 2s_1 + 2$, and let $\mathsf{GUVExt} : \{0,1\}^{\ell} \times \{0,1\}^{d} \to \{0,1\}^{s_1}$ be the $(2s_1, 0.1)$-extractor of Theorem 6, so that $d \leq O(\log\log(1/\delta) + \log S)$. On input $x \in \{0,1\}^n$ and random string $y \in \{0,1\}^{\ell}$, run Algorithm 5.

Algorithm 5: The algorithm used to prove Lemma 36.

```
for z ∈ {0,1}^d do
    Let y_1 ← GUVExt(y, z)
    if Check(P, x, y_1) accepts then  /* Check is the algorithm of Lemma 35 */
        Run B(x, NisGen(y_1, y_2)) for every y_2, take a majority vote, and output the
          answer
    end
end
Output ⊥
```

Clearly, Algorithm 5 runs in space $O(S + d)$. Since δ is constructible in space $O(S)$, its denominator must have at most $2^{O(S)}$ digits. Therefore, $\delta \geq 2^{-2^{O(S)}}$ and $d \leq O(S)$, so the algorithm runs in space $O(S)$. Furthermore, the algorithm is clearly zero-error. Finally, by Proposition 7, the number of y such that $\mathsf{Check}(\mathcal{P}, x, y_1)$ rejects for every z is at most 2^{2s_1+2}, and hence the failure probability of the algorithm is at most $\frac{2^{2s_1+2}}{2^{\ell}} \leq \delta$. ◀

We now give our generalization of Theorem 34. From the work of Goldreich and Wigderson [15], it follows that if a language $L \in \mathbf{BPSPACE}(S)$ is in $\mathbf{DPSPACE}(S)/a$ for $a \ll n$ via an algorithm where most advice strings are "good", then L is close to being in $\mathbf{DPSPACE}(S)$. Our theorem is a *converse*[10] to this result, showing that in the space-bounded setting, there is a very tight connection between typically-correct derandomizations and simulations with small amounts of advice.

▶ **Theorem 37.** *Fix functions $S : \mathbb{N} \to \mathbb{N}$ with $S(n) \geq \log n$ and $\varepsilon : \mathbb{N} \to [0,1]$ that are constructible in space $O(S)$. Suppose a language $L \in \mathbf{BPSPACE}(S)$ is within ε of $\mathbf{DSPACE}(S)$. Then*

$$L \in \mathbf{DSPACE}(S)/(n - \log_2(1/\varepsilon(n)) + O(S^2)). \tag{83}$$

Proof. Let $\mathcal{A}$ be the algorithm of Lemma 36 with $\delta < 2^{-n}/\varepsilon$. Let $m = m(n)$ be the number of random bits used by $\mathcal{A}$. Let $\mathcal{B}$ be the algorithm witnessing the fact that L is within ε of $\mathbf{DSPACE}(S)$.

The algorithm with advice is very simple. Given input $x \in \{0,1\}^n$ and advice $a \in \{0,1\}^m$, output $\mathcal{A}(x,a)$, unless $\mathcal{A}(x,a) = \perp$, in which case output $\mathcal{B}(x)$. This algorithm clearly runs in $O(S)$ space and uses $n - \log_2(1/\varepsilon(n)) + O(S^2)$ bits of advice.

[10] The statement of Theorem 37 doesn't mention it, but indeed, in the proof of Theorem 37, most advice strings are "good".

Now we argue that there is some advice string such that the algorithm succeeds on all inputs. Let $S \subseteq \{0,1\}^n$ be the set of inputs on which $\mathcal{B}$ fails. Consider picking an advice string a uniformly at random. For each string $x \in S$, $\Pr_a[\mathcal{A}(x,a) = \perp] \leq \delta$. Therefore, by the union bound, the probability that there is some $x \in S$ such that $\mathcal{A}(x,a) = \perp$ is at most $|S|\delta = \varepsilon \cdot 2^n \cdot \delta < 1$. Therefore, there is *some* advice string such that the algorithm succeeds on all inputs in S. Finally, for *any* advice string, the algorithm succeeds on all inputs in $\{0,1\}^n \setminus S$, because $\mathcal{A}$ is zero-error. ◀

Combining Theorem 37 with our typically-correct derandomizations gives unconditional simulations with fewer than n bits of advice:

▶ **Corollary 38.** *For every constant $c \in \mathbb{N}$,*

$$\mathbf{BPTISP}(n \operatorname{polylog} n, \log n) \subseteq \mathbf{L}/(n - \log^c n). \tag{84}$$

Proof. Combine Corollary 18 and Theorem 37. ◀

▶ **Corollary 39.** *For every constant $c \in \mathbb{N}$,*

$$\mathbf{BPTISP}_{\mathrm{TM}}(n \operatorname{polylog} n, \log n) \subseteq \mathbf{L}/\left(\frac{n}{\log^c n}\right). \tag{85}$$

Proof. Combine Corollary 23 and Theorem 37. ◀

▶ **Corollary 40.**

$$\mathbf{BPTISP}_{\mathrm{TM}}(n^{1.99}, \log n) \subseteq \mathbf{L}/(n - n^{\Omega(1)}). \tag{86}$$

Proof. Combine Corollary 33 and Theorem 37. ◀

6 Disambiguating Efficient Nondeterministic Algorithms

6.1 Overview

Recall that a nondeterministic algorithm is *unambiguous* if on every input, there is at most one accepting computation. Suppose a language L can be decided by a nondeterministic algorithm that runs in time $T = T(n) \geq n$ and space $S = S(n) \geq \log n$. Allender, Reinhardt, and Zhou showed that if SAT has exponential circuit complexity, there is an unambiguous algorithm for L that runs in space $O(S)$ [2]. Unconditionally, van Melkebeek and Prakriya recently gave an unambiguous algorithm for L that runs in time $2^{O(S)}$ and space $O(S\sqrt{\log T})$ [40].

For some of our results on derandomizing efficient algorithms, we give a corresponding theorem for disambiguating efficient nondeterministic algorithms, albeit with slightly worse parameters.

6.1.1 Our Results

Let $\mathbf{NTISP}(T,S)$ denote the class of languages that can be decided by a nondeterministic random-access Turing machines that runs in time T and space S. Define $\mathbf{UTISP}(T,S)$ the same way, but with the additional requirement that the algorithm is unambiguous. In Sections 6.4 and 6.5, we show that for every S and every constant $c \in \mathbb{N}$,

$$\mathbf{NTISP}(n \cdot \operatorname{poly}(S), S) \text{ is within } 2^{-S^c} \text{ of } \mathbf{UTISP}(2^{O(S)}, S\sqrt{\log S}). \tag{87}$$

Equation (87) is analogous to Corollary 18.

Reinhardt and Allender showed that $\mathbf{NL} \subseteq \mathbf{UL}/\operatorname{poly}$ [31]. In Section 6.6, we improve the Reinhardt-Allender theorem by showing that $\mathbf{NL} \subseteq \mathbf{UL}/(n + O(\log^2 n))$. More generally, we show that if a language $L \in \mathbf{NSPACE}(S)$ is within $\varepsilon(n)$ of being in $\mathbf{USPACE}(S)$, then $L \in \mathbf{USPACE}(S)/(n - \log_2(1/\varepsilon(n)) + O(S^2))$. This result is analogous to Theorem 37.

6.1.2 Techniques

Our disambiguation theorems are proven using the same "out of sight, out of mind" technique that we used in Sections 3 and 4.2 for derandomization. Roughly, this is possible because of prior work [31, 40] that reduces the problem of disambiguating algorithms to certain derandomization problems. We review the necessary background in Section 6.3.

Our disambiguation algorithms do not really introduce any additional novel techniques, beyond what we already used in Sections 3 and 4.2. Rather, our contribution in this section is to identify another setting where our techniques are helpful, thereby illustrating the generality of our techniques.

6.2 Preliminaries

Unambiguous algorithms can be *composed* as long as the inner algorithm is "single-valued", which we now define. This notion corresponds to classes such as $\mathbf{UL} \cap \mathbf{coUL}$.

▶ **Definition 41.** *A* single-valued unambiguous algorithm $\mathcal{A}$ *is a nondeterministic algorithm such that for every input x, all but one computation path outputs a special symbol $\perp_{\mathrm{n}}$ (indicating that the nondeterministic choices were "bad"). We let $\mathcal{A}(x)$ denote the output of the one remaining computation path.*

When describing unambiguous algorithms, we will often include steps such as "Compute $a \leftarrow \mathcal{A}(x)$", where $\mathcal{A}$ is a single-valued unambiguous algorithm. Such a step should be understood as saying to run $\mathcal{A}$ on input x. If $\mathcal{A}$ outputs $\perp_{\mathrm{n}}$, immediately halt and output $\perp_{\mathrm{n}}$. Otherwise, let a be the output of $\mathcal{A}$.

6.3 Unambiguous Algorithms for Connectivity by van Melkebeek and Prakriya

Recall that the *s-t connectivity problem* is defined by

$$\mathrm{STConn} = \{(G, s, t) : \text{there is a directed path from } s \text{ to } t\}, \tag{88}$$

where G is a digraph and $s, t \in V(G)$. STConn is a classic example of an **NL**-complete language [21]. Using an "inductive counting" technique, Reinhardt and Allender gave a single-valued unambiguous algorithm for testing whether a given digraph is "min-unique", as well as a single-valued unambiguous algorithm for solving STConn in min-unique digraphs [31]. Using the isolation lemma, Reinhardt and Allender showed that assigning random weights to a digraph makes it "min-unique" [31]. These two results are the main ingredients in the proof that $\mathbf{NL} \subseteq \mathbf{UL}/\operatorname{poly}$ [31].

Recently, van Melkebeek and Prakriya gave a "pseudorandom weight generator" with seed length $O(\log^2 n)$ [40].[11] Just like uniform random weights, the weights produced by this generator make a digraph "min-unique" with high probability.[12]

[11] In the terminology of van Melkebeek and Prakriya [40], here we refer to the "hashing only" approach.
[12] The van Melkebeek-Prakriya generator only works for *layered* digraphs, but this technicality does not matter for us.

Roughly, this pseudorandom weight generator by van Melkebeek and Prakriya will play a role in our disambiguation results that is analogous to the role that Nisan's generator played in our derandomization results.

For our purposes, it is not necessary to give a precise account of min-uniqueness. What matters is that STConn can be decided in unambiguous log-space given two-way access to an $O(\log^2 n)$-bit random string. Furthermore, "bad" random strings can be unambiguously detected. We now state this result more carefully.

▶ **Theorem 42** ([40]). *There is a single-valued unambiguous algorithm* vMPSeededAlg *so that for every* $x \in \{0,1\}^n$,

$$\Pr_{y \in \{0,1\}^\infty}[\mathsf{vMPSeededAlg}(x,y) \in \{\mathrm{STConn}(x), \perp_{\mathrm{r}}\}] = 1, \tag{89}$$

$$\Pr_{y \in \{0,1\}^\infty}[\mathsf{vMPSeededAlg}(x,y) = \perp_{\mathrm{r}}] \leq 1/2. \tag{90}$$

Furthermore, $\mathsf{vMPSeededAlg}(x,y)$ *only reads the first* $O(\log^2 n)$ *bits of* y *(the "seed") and runs in space* $O(\log n)$.

Proof sketch. We assume that the reader is familiar with the paper by van Melkebeek and Prakriya [40]. Given an instance x of STConn, the algorithm vMPSeededAlg first applies a reduction, giving a *layered* digraph G on which to test connectivity. Then, the first $O(\log^2 n)$ bits of y are interpreted as specifying $O(\log n)$ hash functions, which are used to assign weights to the vertices in G. An algorithm by Reinhardt and Allender [31] is run to determine whether the resulting weighted digraph is min-unique. If it is not, vMPSeededAlg outputs $\perp_{\mathrm{r}}$. If it is, another closely related algorithm by Reinhardt and Allender [31] is run to decide connectivity in the resulting weighted digraph. ◀

Notice that vMPSeededAlg can be thought of as having *three* read-only inputs: the "real" input $x \in \{0,1\}^n$; the random seed $y \in \{0,1\}^{O(\log^2 n)}$; and the nondeterministic bits $z \in \{0,1\}^{\mathrm{poly}(n)}$. The algorithm has two-way access to x and y and one-way access to z. Notice also that a computation path of vMPSeededAlg has *four* possible outputs: 0, indicating that $x \notin$ STConn; 1, indicating that $x \in$ STConn; $\perp_{\mathrm{n}}$, indicating bad nondeterministic bits z; and $\perp_{\mathrm{r}}$, indicating bad random bits y.

Iterating over all y in Theorem 42 would take $\Theta(\log^2 n)$ space. By modifying their "pseudorandom weight generator", van Melkebeek and Prakriya gave an unambiguous algorithm for STConn that runs in $O(\log^{3/2} n)$ space. The performance of their algorithm is improved if we only need to search for *short* paths; the precise details are given by the following theorem.

▶ **Theorem 43** ([40]). *There is a single-valued unambiguous algorithm* vMPShortPathsAlg *such that if* G *is a digraph,* $s,t \in V(G)$*, and* $r \in \mathbb{N}$*, then* $\mathsf{vMPShortPathsAlg}(G,s,t,r) = 1$ *if and only if there is a directed path from* s *to* t *in* G *of length at most* r*. Furthermore,* vMPShortPathsAlg *runs in time* $\mathrm{poly}(n)$ *and space* $O(\log n \sqrt{\log r})$.

Proof sketch. Again, we assume that the reader is familiar with the paper by van Melkebeek and Prakriya [40]. Again, we first apply a reduction, giving a layered digraph G' of width $|V(G)|$ and length r, so that the question is whether there is a path from the first vertex in the first layer to the first vertex in the last layer.

We rely on the "combined hashing and shifting" generator by van Melkebeek and Prakriya [40, Theorem 1]. The seed of this generator specifies $O(\sqrt{\log r})$ hash functions (each is specified with $O(\log n)$ bits). We find these hash functions by exhaustive search one at a time, maintaining the invariant that portions of G' that have weights assigned are min-unique. We test for min-uniqueness using a slight variant of the algorithm by Reinhardt and Allender [31] described by van Melkebeek and Prakriya [40, Lemma 1]. ◀

Roughly speaking, Theorem 43 plays a role in our disambiguation results that is analogous to the role that the Nisan-Zuckerman generator played in our derandomization results.

6.4 Disambiguating Branching Programs

For us, a *nondeterministic branching program* $\mathcal{P}$ on $\{0,1\}^n \times \{0,1\}^m$ is a randomized branching program (but we think of the second input to the program as nondeterministic bits instead of random bits) such that some vertex $v_0 \in V(\mathcal{P})$ is labeled as the *start vertex* and some vertex $v_{\text{accept}} \in V(\mathcal{P})$ is labeled as the *accepting vertex*. We identify $\mathcal{P}$ with a function $\mathcal{P} : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}$ defined by

$$\mathcal{P}(x,y) = \begin{cases} 1 & \text{if } \mathcal{P}(v_0; x, y) = v_{\text{accept}}, \\ 0 & \text{otherwise,} \end{cases} \tag{91}$$

and we *also* identify $\mathcal{P}$ with a function $\mathcal{P} : \{0,1\}^n \to \{0,1\}$ defined by

$$\mathcal{P}(x) = 1 \iff \exists y\, \mathcal{P}(x,y) = 1. \tag{92}$$

(Equation (92) expresses the fact that $\mathcal{P}$ is a *nondeterministic* branching program.) Finally, an *R-OW* nondeterministic branching program is just a nondeterministic branching program that is R-OW when thought of as a randomized branching program, i.e., it reads its nondeterministic bits from left to right.

▶ **Theorem 44.** *For every constant $c \in \mathbb{N}$, there is a single-valued unambiguous algorithm* A *with the following properties. Suppose $\mathcal{P}$ is an R-OW nondeterministic branching program on $\{0,1\}^n \times \{0,1\}^T$. Suppose $S \geq \log n$, where $S \stackrel{\text{def}}{=} \lceil \log \text{size}(\mathcal{P}) \rceil$, and $T \geq \text{length}(\mathcal{P})$. Then*

$$\text{density}\{x \in \{0,1\}^n : \mathsf{A}(\mathcal{P}, x, T) \neq \mathcal{P}(x)\} \leq 2^{-S^c}. \tag{93}$$

Furthermore, $\mathsf{A}(\mathcal{P}, x, T)$ runs in time $2^{O(S)}$ and space $O(S\sqrt{\log\lceil T/n \rceil + \log S})$.

Toward proving Theorem 44, we introduce some notation. The computation of $\mathcal{P}(x)$ naturally reduces to STConn. Let $\mathcal{P}[x]$ be the digraph (V, E), where $V = V(\mathcal{P})$ and E is the set of edges (u, v) in $\mathcal{P}$ labeled with $x_{i(u)}0$ or $x_{i(u)}1$. (So every nonterminal vertex in $\mathcal{P}[x]$ has outdegree 2.) That way, $\mathcal{P}(x) = 1$ if and only if $(\mathcal{P}[x], v_0, v_{\text{accept}}) \in \text{STConn}$.

The algorithm A of Theorem 44 is given in Algorithm 6. The algorithm relies on a subroutine B given in Algorithm 7.

Parameters

Let s be the number of random bits used by vMPSeededAlg, so that $s \leq O(S^2)$. The subroutine B relies on the extractor GUVExt of Theorem 6. This extractor is instantiated with source length $\ell \stackrel{\text{def}}{=} S^{c+1}$, error 0.1, entropy $k \stackrel{\text{def}}{=} 2s$, and output length s. The seed length of GUVExt is $d \leq O(\log \ell) = O(\log S)$.

Efficiency

First, we bound the space complexity of A. If $S^{c+1} > n$, then A runs in space

$$O(\log \text{size}(\mathcal{P})\sqrt{\log T}) = O(S\sqrt{\log T}) \leq O\left(S\sqrt{\log \frac{TS^{c+1}}{n}}\right) \tag{94}$$

$$= O(S\sqrt{\log(T/n) + \log S}). \tag{95}$$

Algorithm 6: The algorithm A of Theorem 44.

if $S^{c+1} > n$ **then**
  return vMPShortPathsAlg$(\mathcal{P}[x], v_0, v_{accept}, T)$
else
  Let $I_1, I_2, \ldots, I_B \subseteq [n]$ be disjoint sets of size S^{c+1} with B as large as possible
  for $b = 1$ **to** B **do**
    Let $I \leftarrow I_b$
    Let $V_b \leftarrow \{v \in V(\mathcal{P}) : i(v) \in I\} \cup \{v_0, v_{\text{accept}}\}$
    Let E_b be the set of pairs $(u, v) \in V_b^2$ such that there is a directed path from u to v in $\mathcal{P}|_{[n]\setminus I}[x]$
    Let H_b be the digraph (V_b, E_b)
    Compute $a \leftarrow$ vMPShortPathsAlg$(H_b, v_0, v_{\text{accept}}, \lfloor S^{c+1}T/n \rfloor + 1)$. Whenever vMPShortPathsAlg asks whether some pair (u, v) is in E_b, run B$(\mathcal{P}, x, b, u, v)$, where B is given in Algorithm 7
    if $a = 1$ **then return** 1
  end
  return 0
end

Algorithm 7: The algorithm B used by A to decide whether $(u, v) \in E_b$. The block I is the same block I_b used by A.

for $y \in \{0,1\}^{O(\log S)}$ **do**
  Let $a' \leftarrow$ vMPSeededAlg$(\mathcal{P}|_{[n]\setminus I}[x], u, v,$ GUVExt$(x|_I, y))$
  if $a' \neq \perp_{\text{r}}$ **then return** a'
end
return $\perp_{\text{i}}$

Suppose now that $S^{c+1} \leq n$. The extractor GUVExt runs in space $O(\log S)$, and the algorithm vMPSeededAlg runs in space $O(S)$, so B runs in space $O(S)$. The algorithm vMPShortPathsAlg runs in space

$$O(\log |V_b| \sqrt{\log(\lfloor S^{c+1}T/n \rfloor + 1)}) \leq O(S\sqrt{\log \lceil T/n \rceil + \log S}). \tag{96}$$

Therefore, overall, A runs in space $O(S\sqrt{\log \lceil T/n \rceil + \log S})$.

Next, we bound the running time of A. If $S^{c+1} > n$, then A runs in time $\text{poly}(\text{size}(\mathcal{P})) = 2^{O(S)}$ as claimed. Suppose now that $S^{c+1} \leq n$. Because B runs in space $O(S)$, it must run in time $2^{O(S)}$. Therefore, vMPShortPathsAlg runs in time $2^{O(S)} \cdot 2^{O(S)} = 2^{O(S)}$. Therefore, overall, A runs in time $2^{O(S)}$.

Correctness

Since vMPSeededAlg and vMPShortPathsAlg are single-valued unambiguous algorithms, A is a single-valued unambiguous algorithm. All that remains is to show that for most x, $\text{A}(\mathcal{P}, x, T) = \mathcal{P}(x)$. First, we show that for most x, the subroutine B is correct, i.e., the one computation path that does not output $\perp_{\text{n}}$ outputs a bit indicating whether $(u, v) \in E_b$. Clearly, the only way that B can be incorrect is if it outputs $\perp_{\text{i}}$, indicating a "hard" input x.

▷ Claim 45. For every $\mathcal{P}$,

$$\text{density}\{x \in \{0,1\}^n : \exists b, u, v \text{ such that } \mathsf{B}(\mathcal{P}, x, b, u, v) = \perp_{\mathrm{i}}\} \leq 2^{-S^c}. \tag{97}$$

Proof. The graph $\mathcal{P}|_{[n]\setminus I}[x]$ does not depend on $x|_I$. Therefore, for each fixed b, each fixed $z \in \{0,1\}^{n-|I_b|}$, and each fixed $u, v \in V(\mathcal{P})$, by Proposition 7,

$$\#\{x : x|_{[n]\setminus I} = z \text{ and } \mathsf{B}(\mathcal{P}, x, b, u, v) = \perp_{\mathrm{i}}\} \leq 2^{k+2} \leq 2^{O(S^4)}. \tag{98}$$

Therefore, by summing over all b, z, u, v,

$$\begin{aligned}\#\{x \in \{0,1\}^n : \exists b, u, v \text{ such that } \mathsf{B}(\mathcal{P}, x, b, u, v) = \perp_{\mathrm{i}}\} &\leq 2^{n - S^{c+1} + \log n + 2S + O(S^4)} && (99)\\ &= 2^{n - S^{c+1} + O(S^4)} && (100)\\ &\leq 2^{n - S^c} && (101)\end{aligned}$$

for sufficiently large n. ◁

Next, we show that as long as B does not make any mistakes, A is correct.

▷ Claim 46. If $\mathcal{P}(x) = 1$, there is some $b \in [B]$ so that there is a path from v_0 to v_{accept} through H_b of length at most $\lfloor S^{c+1}T/n \rfloor + 1$.

Proof. Since $\mathcal{P}(x) = 1$, there is a path from v_0 to v_{accept} through $\mathcal{P}[x]$ of length at most T. Let $v_0, v_1, v_2, \ldots, v_{T'} = v_{\text{accept}}$ be the vertices visited by that path, so that $T' \leq T$. Consider picking $b \in [B]$ uniformly at random. Then for each $t < T'$, $\Pr[i(v_t) \in I_b] \leq S^{c+1}/n$. Therefore, by linearity of expectation,

$$\mathrm{E}[\#\{t : i(v_t) \in I_b\}] \leq S^{c+1}T/n. \tag{102}$$

The best case is at least as good as the average case, so there is some $b \in [B]$ such that $\#\{t : i(v_t) \in I_b\} \leq S^{c+1}T/n$. Let $t_1, t_2, \ldots, t_r$ be the indices t such that $i(v_t) \in I_b$. Then by the definition of E_b, the edges $(v_0, t_1), (t_1, t_2), \ldots, (t_{r-1}, t_r), (t_r, v_{\text{accept}})$ are all present in H_b. Therefore, there is a path from v_0 to v_{accept} through H_b of length at most $r + 1$. ◁

Combining Claims 45 and 46 completes the proof of Theorem 44.

6.5 Disambiguating Uniform Random-Access Algorithms

▶ **Corollary 47.** *For every space-constructible function $S(n) \geq \log n$, for every constant $c \in \mathbb{N}$,*

$$\mathbf{NTISP}(n \cdot \mathrm{poly}(S), S) \text{ is within } 2^{-S^c} \text{ of } \mathbf{UTISP}(2^{O(S)}, S\sqrt{\log S}). \tag{103}$$

Proof sketch. The class $\mathbf{NTISP}(n \cdot \mathrm{poly}(S), S)$ corresponds to R-OW nondeterministic branching programs of size $2^{O(S)}$ and length $T = n \cdot \mathrm{poly}(S)$. For these parameters, the algorithm of Theorem 44 runs in time $2^{O(S)}$ and space $O(S\sqrt{\log S})$. ◀

6.6 Disambiguation with Advice

We now show how to disambiguate $\mathbf{NL}$ with only $n + O(\log^2 n)$ bits of advice. The proof is very similar to the proof of Theorem 34.

▶ **Theorem 48.** $\mathbf{NL} \subseteq \mathbf{UL}/(n + O(\log^2 n))$.

Proof. Let $\mathcal{R}$ be a log-space reduction from $L \in \mathbf{NL}$ to STConn. Let s be the number of random bits used by $\mathsf{vMPSeededAlg}$ on inputs of length n^c, where n^c is the length of outputs of $\mathcal{R}$ on inputs of length n. Let $\mathsf{GUVExt} : \{0,1\}^{n+2s+3} \times \{0,1\}^d \to \{0,1\}^s$ be the $(2s, 0.1)$-extractor of Theorem 6, so that $d \leq O(\log n)$.

Given input $x \in \{0,1\}^n$ and advice $a \in \{0,1\}^{n+2s+3}$, compute

$$a_z \stackrel{\text{def}}{=} \mathsf{vMPSeededAlg}(\mathcal{R}(x), \mathsf{GUVExt}(a, z)) \tag{104}$$

for all z and accept if there is some z so that $a_z = 1$.

This algorithm clearly runs in space $O(\log n)$ and is unambiguous. By Proposition 7, for each fixed x, the number of advice strings a causing the algorithm to give the wrong answer is at most 2^{2s+2}. Therefore, the total number of advice strings a that cause the algorithm to give the wrong answer for any x is at most $2^{n+2s+2} < 2^{|a|}$. Therefore, there is some choice of a such that the algorithm succeeds on all inputs. ◀

Just like we did with Theorem 34, we now generalize Theorem 48, showing that the amount of advice can be reduced to below n if we start with a language that has a typically-correct disambiguation.

▶ **Theorem 49.** *Fix functions $S : \mathbb{N} \to \mathbb{N}$ with $S(n) \geq \log n$ and $\varepsilon : \mathbb{N} \to [0,1]$ that are constructible in $O(S)$ space. Suppose a language $L \in \mathbf{NSPACE}(S)$ is within ε of $\mathbf{USPACE}(S)$. Then*

$$L \in \mathbf{USPACE}(S)/(n - \log_2(1/\varepsilon(n)) + O(S^2)). \tag{105}$$

The proof of Theorem 49 is very similar to the proof of Theorem 37. Because the proof of Theorem 49 does not introduce any significantly new techniques, we defer the proof to Appendix E.

▶ **Corollary 50.** *For every constant $c \in \mathbb{N}$,*

$$\mathbf{NTISP}(n \operatorname{polylog} n, \log n) \subseteq \mathbf{USPACE}(\log n \sqrt{\log \log n})/(n - \log^c n). \tag{106}$$

Proof. For any $L \in \mathbf{NTISP}(n \operatorname{polylog} n, \log n)$, obviously $L \in \mathbf{NSPACE}(\log n \sqrt{\log \log n})$, and by Corollary 47, L is within $2^{-\log^c n}$ of $\mathbf{USPACE}(\log n \sqrt{\log \log n})$. Applying Theorem 49 completes the proof. ◀

7 Directions for Further Research

The main open problem in this area is to prove that $\mathbf{BPL}$ is within $o(1)$ of $\mathbf{L}$. Corollary 18 implies that $\mathbf{BPTISP}(n \operatorname{polylog} n, \log n)$ is within $o(1)$ of $\mathbf{L}$, and Corollary 33 implies that $\mathbf{BPTISP}_{\mathrm{TM}}(n^{1.99}, \log n)$ is within $o(1)$ of $\mathbf{L}$, but $\mathbf{BPL}$ allows time n^c where c is an arbitrarily large constant. At present, for a generic language $L \in \mathbf{BPL}$, we do not even know a deterministic log-space algorithm that succeeds on at least *one* input of each length.

This work also provides some additional motivation for studying small-space extractors. The two extractors we used in this paper (Theorems 5 and 6) were sufficient for our applications, but it would be nice to have a single log-space extractor that is optimal up to constants for the full range of parameters.

References

1 Leonard Adleman. Two theorems on random polynomial time. In *Proceedings of the 19th Annual Symposium on Foundations of Computer Science (FOCS '78)*, pages 75–83. IEEE, 1978.

2 Eric Allender, Klaus Reinhardt, and Shiyu Zhou. Isolation, matching, and counting uniform and nonuniform upper bounds. *Journal of Computer and System Sciences*, 59(2):164–181, 1999. doi:10.1006/jcss.1999.1646.

3 Josh Alman. An illuminating algorithm for the light bulb problem. In *2nd Symposium on Simplicity in Algorithms*, volume 69 of *OASIcs OpenAccess Ser. Inform.*, pages Art. No. 2, 11. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2019.

4 Roy Armoni. On the derandomization of space-bounded computations. In *Proceedings of the 2nd International Workshop on Randomization and Computation (RANDOM '98)*, volume 1518 of *Lecture Notes in Computer Science*, pages 47–59. Springer, Berlin, 1998. doi:10.1007/3-540-49543-6_5.

5 Vikraman Arvind and Jacobo Toran. Solvable group isomorphism is (almost) in NP ∩ coNP. In *Proceedings of the 19th Annual Conference on Computational Complexity (CCC '04)*, pages 91–103. IEEE, 2004.

6 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.

7 László Babai, Noam Nisan, and Márió Szegedy. Multiparty protocols, pseudorandom generators for logspace, and time-space trade-offs. *Journal of Computer and System Sciences*, 45(2):204–232, 1992. doi:10.1016/0022-0000(92)90047-M.

8 Paul Beame, Michael Saks, Xiaodong Sun, and Erik Vee. Time-space trade-off lower bounds for randomized computation of decision problems. *Journal of the ACM*, 50(2):154–195, 2003. doi:10.1145/636865.636867.

9 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.

10 Jin-Yi Cai, Venkatesan T. Chakaravarthy, and Dieter van Melkebeek. Time-space tradeoff in derandomizing probabilistic logspace. *Theory of Computing Systems*, 39(1):189–208, 2006. doi:10.1007/s00224-005-1264-9.

11 Scott Diehl and Dieter van Melkebeek. Time-space lower bounds for the polynomial-time hierarchy on randomized machines. *SIAM Journal on Computing*, 36(3):563–594, 2006. doi:10.1137/050642228.

12 Lance Fortnow and Adam R. Klivans. Linear advice for randomized logarithmic space. In *Proceedings of the 23rd Annual Symposium on Theoretical Aspects of Computer Science (STACS '06)*, volume 3884 of *Lecture Notes in Computer Science*, pages 469–476. Springer, Berlin, 2006. doi:10.1007/11672142_38.

13 Lance Fortnow and Dieter van Melkebeek. Time-space tradeoffs for nondeterministic computation. In *Proceedings of the 15th Annual Conference on Computational Complexity (CCC '00)*, pages 2–13. IEEE, 2000. doi:10.1109/CCC.2000.856730.

14 David Gillman. A Chernoff bound for random walks on expander graphs. *SIAM Journal on Computing*, 27(4):1203–1220, 1998. doi:10.1137/S0097539794268765.

15 Oded Goldreich and Avi Wigderson. Derandomization that is rarely wrong from short advice that is typically good. In *Randomization and approximation techniques in computer science (RANDOM '02)*, volume 2483 of *Lecture Notes in Computer Science*, pages 209–223. Springer, Berlin, 2002. doi:10.1007/3-540-45726-7_17.

16 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. doi:10.1145/1538902.1538904.

17 Dan Gutfreund and Emanuele Viola. Fooling parity tests with parity gates. In *Proceedings of the 8th International Workshop on Randomization and Computation (RANDOM '04)*, volume 3122 of *Lecture Notes in Computer Science*, pages 381–392. Springer, 2004.

18 Tzvika Hartman and Ran Raz. On the distribution of the number of roots of polynomials and explicit weak designs. *Random Structures & Algorithms*, 23(3):235–263, 2003. doi:10.1002/rsa.10095.

19 Lane A Hemaspaandra and Ryan Williams. SIGACT news complexity theory column 76: an atypical survey of typical-case heuristic algorithms. *ACM SIGACT News*, 43(4):70–89, 2012.

20 Russell Impagliazzo and Avi Wigderson. P = BPP if E requires exponential circuits: Derandomizing the XOR lemma. In *Proceedings of the 29th Annual Symposium on Theory of Computing (STOC '97)*, pages 220–229, New York, NY, USA, 1997. ACM. doi:10.1145/258533.258590.

21 Neil D. Jones. Space-bounded reducibility among combinatorial problems. *Journal of Computer and System Sciences*, 11(1):68–85, 1975. doi:10.1016/S0022-0000(75)80050-X.

22 Daniel M Kane, Jelani Nelson, and David P Woodruff. Revisiting norm estimation in data streams. *arXiv preprint*, 2008. arXiv:0811.3648.

23 Neeraj Kayal and Nitin Saxena. On the ring isomorphism & automorphism problems. In *Proceedings of the 20th Annual Conference on Computational Complexity (CCC '05)*, pages 2–12. IEEE, 2005.

24 Jeff Kinne, Dieter van Melkebeek, and Ronen Shaltiel. Pseudorandom generators, typically-correct derandomization, and circuit lower bounds. *Computational Complexity*, 21(1):3–61, 2012. doi:10.1007/s00037-011-0019-z.

25 Adam R. Klivans and Dieter van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. *SIAM Journal on Computing*, 31(5):1501–1526, 2002. doi:10.1137/S0097539700389652.

26 Noam Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992. doi:10.1007/BF01305237.

27 Noam Nisan. On read-once vs. multiple access to randomness in logspace. *Theoretical Computer Science*, 107(1):135–144, 1993. doi:10.1016/0304-3975(93)90258-U.

28 Noam Nisan. RL $\subseteq$ SC. *Computational Complexity*, 4(1):1–11, 1994. doi:10.1007/BF01205052.

29 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.

30 Noam Nisan and David Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, 52(1):43–52, 1996. doi:10.1006/jcss.1996.0004.

31 Klaus Reinhardt and Eric Allender. Making nondeterminism unambiguous. *SIAM Journal on Computing*, 29(4):1118–1131, 2000. doi:10.1137/S0097539798339041.

32 Michael Saks and Shiyu Zhou. $BP_HSPACE(S) \subseteq DSPACE(S^{3/2})$. *Journal of Computer and System Sciences*, 58(2):376–403, 1999.

33 Rahul Santhanam and Ryan Williams. On uniformity and circuit lower bounds. *Computational Complexity*, 23(2):177–205, 2014. doi:10.1007/s00037-014-0087-y.

34 Ronen Shaltiel. Typically-correct derandomization. *ACM SIGACT News*, 41(2):57–72, 2010.

35 Ronen Shaltiel. Weak derandomization of weak algorithms: explicit versions of Yao's lemma. *Computational Complexity*, 20(1):87–143, 2011. doi:10.1007/s00037-011-0006-4.

36 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:10.1145/1059513.1059516.

37 Madhu Sudan, Luca Trevisan, and Salil Vadhan. Pseudorandom generators without the XOR lemma. *Journal of Computer and System Sciences*, 62(2):236–266, 2001. Special issue on the 14th Annual Conference on Computational Complexity (CCC '99). doi:10.1006/jcss.2000.1730.

38 Salil P. Vadhan. Pseudorandomness. *Foundations and Trends® in Theoretical Computer Science*, 7(1–3):1–336, 2012. doi:10.1561/0400000010.

39 J. H. van Lint. *Introduction to coding theory*, volume 86 of *Graduate Texts in Mathematics*. Springer-Verlag, Berlin, third edition, 1999. doi:10.1007/978-3-642-58575-3.

40 Dieter van Melkebeek and Gautam Prakriya. Derandomizing Isolation in Space-Bounded Settings. In *32nd Annual Conference on Computational Complexity (CCC '17)*, volume 79 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 5:1–5:32, Dagstuhl, Germany, 2017. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.CCC.2017.5.

41 Dieter van Melkebeek and Rahul Santhanam. Holographic proofs and derandomization. *SIAM Journal on Computing*, 35(1):59–90, 2005. doi:10.1137/S0097539703438629.

42 Andrew C. Yao. Theory and applications of trapdoor functions. In *23rd Annual Symposium on Foundations of Computer Science (FOCS '82)*, pages 80–91. IEEE, New York, 1982.

43 Marius Zimand. Exposure-resilient extractors and the derandomization of probabilistic sublinear time. *Computational Complexity*, 17(2):220–253, 2008. doi:10.1007/s00037-008-0243-3.

44 David Zuckerman. Randomness-optimal oblivious sampling. *Random Structures & Algorithms*, 11(4):345–367, 1997. doi:10.1002/(SICI)1098-2418(199712)11:4<345::AID-RSA4>3.0.CO;2-Z.

A Proof of Theorem 5: The Shaltiel-Umans Extractor

In this section, we discuss the proof of Theorem 5. The extractor follows the same basic construction that Shaltiel and Umans used for a "low error" extractor [36, Corollary 4.21]. We will assume that the reader is familiar with the paper by Shaltiel and Umans [36]. We will also switch to the parameter names by Shaltiel and Umans, so the source length of the extractor is n rather than ℓ, and the seed length is t rather than d. In these terms, we are shooting for time $\mathrm{poly}(n)$ and space $O(t)$.

The only change to the *construction* that we make is that we will use a different instantiation of the "base field" $\mathbb{F}_q$. Shaltiel and Umans [36] used a deterministic algorithm by Shoup that finds an irreducible polynomial of degree $\log q$ over $\mathbb{F}_2$ in time $\mathrm{poly}(\log q)$. Unfortunately, Shoup's algorithm is not sufficiently space-efficient for our purposes. To get around this issue, we use an extremely explicit family of irreducible polynomials:

▶ **Lemma 51** ([39, Theorem 1.1.28]). *For every $a \in \mathbb{N}$, the polynomial $x^{2\cdot 3^a} + x^{3^a} + 1$ is irreducible over $\mathbb{F}_2$.*

Therefore, by replacing q by some power of two between q and q^3, we can easily, deterministically construct an irreducible polynomial of degree $\log q$ in time $\mathrm{poly}(\log q)$ and space $O(\log q)$. This only affects the bit length of field elements, $\log q$, by at most a factor of 3. Therefore, the hypotheses of Shaltiel and Umans' main technical theorem [36, Theorem 4.5] are still met, so the extractor is still correct.

Now we turn to analyzing the efficiency of the extractor. The parameters h, d, m, ρ, q used by Shaltiel and Umans (with the described modification to q) can all easily be computed in time $\mathrm{poly}(n)$ and space $O(t)$. Next, we inspect the construction of the matrix B used by Shaltiel and Umans [36, Proof of Lemma 4.18]. The exhaustive search used to find the irreducible polynomial $p(z)$ takes space $O(d \log q) \le O(t)$. The exhaustive search used to find the generator g for $(H^d)^\times$ also takes space $O(d \log q) = O(t)$. Finally, multiplication by g takes space $O(d \log q) = O(t)$.

It follows immediately that the "q-ary extractor" E' given by Shaltiel and Umans [36, Equation 8] runs in space $O(t)$, because we only need to store the vector $B^i \vec{v}$. Finally, to get from E' to the final extractor, a simple Hadamard code is applied, which can trivially be computed in time $\mathrm{poly}(n)$ and space $O(t)$.

B Proof of Theorem 6: The GUV Extractor

In this section, we discuss the proof of Theorem 6. We will assume that the reader is familiar with the paper by Guruswami, Umans, and Vadhan. Recall that a *condenser* is like an extractor, except that the output is merely guaranteed to be close to having high entropy instead of being guaranteed to be close to uniform.

▶ **Definition 52.** *A function* $\mathsf{Con} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{n'}$ *is a* $k \to_\varepsilon k'$ condenser *if for every random variable* X *with* $H_\infty(X) \geq k$*, there exists a distribution* Z *with* $H_\infty(Z) \geq k'$ *such that if we let* $Y \sim U_d$ *be independent of* X*, then* $\mathsf{Con}(X,Y) \sim_\varepsilon Z$*.*

Guruswami, Umans, and Vadhan constructed a lossy condenser based on folded Reed-Solomon codes [16, Theorem 6.2]. To ensure space efficiency, we will slightly modify their construction to get the following condenser. We will follow the parameter names by Guruswami, Umans, and Vadhan.

▶ **Theorem 53** (Based on [16, Theorem 6.2])**.** *Let* $\alpha > 0$ *be a constant. Consider any* $n \in \mathbb{N}, \ell \leq n$ *such that* 2^ℓ *is an integer and any* $\varepsilon > 0$*. There is a parameter* $t = \Theta(\log(n\ell/\varepsilon))$ *and a*

$$(1 + 1/\alpha)\ell t + \log(1/\varepsilon) \to_{3\varepsilon} \ell t + d - 2$$

condenser $\mathsf{GUVCon} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{n'}$*, computable in space* $O(d)$*, with seed length* $d \leq (1 + 1/\alpha)t$ *and output length* $n' \leq (1 + 1/\alpha)\ell t + d$*, provided* $\ell t \geq \log(1/\varepsilon)$*.*

Proof sketch. We need to use a base field $\mathbb{F}_q$ based on Lemma 51, so we slightly modify the parameters of the GUV construction as follows. Choose q to be the smallest power of two of the form $2^{2 \cdot 3^a}$ such that $q \geq (2^{2+1/\alpha} \cdot n\ell/\varepsilon)^{1+\alpha}$. This q satisfies $q \leq (2^{2+1/\alpha} \cdot n\ell/\varepsilon)^{3+3\alpha}$. Next, define $t = \lceil \frac{\alpha \log q}{1+\alpha} \rceil$ and $h = 2^t$, so that $q \in ((h/2)^{1+1/\alpha}, h^{1+1/\alpha}]$. Therefore, we still have

$$\begin{aligned} q &> h \cdot h^{1/\alpha}/2^{1+1/\alpha} && (107)\\ &\geq h \cdot q^{1/(1+\alpha)}/2^{1+1/\alpha} && (108)\\ &\geq 2hn\ell/\varepsilon, && (109) \end{aligned}$$

and hence $A \geq \varepsilon q/2$. The rest of the argument is as in the original paper [16]. ◀

There is a standard extractor based on expander walks that works well for constant error and constant entropy rate. Using the Gutfreund-Viola expander walk (Theorem 31), this extractor runs in logarithmic space:

▶ **Lemma 54.** *Let* $\alpha, \varepsilon > 0$ *be constants. There is some constant* $\beta \in (0,1)$ *so that for all* n*, there is a* $(\beta n, \varepsilon)$*-extractor* $\mathsf{GVExt} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *with* $t \leq \log(\alpha n)$ *and* $m \geq (1-\alpha)n$ *so that given* x *and* y*,* $\mathsf{GVExt}(x,y)$ *can be computed in* $O(\log n)$ *space.*

Proof sketch. This construction of an extractor from an expander is standard; see, e.g., an exposition by Guruswami et al. [16, Theorem 4.6]. The space bound follows from Theorem 31. ◀

Finally, Theorem 6 follows by composing Theorem 53 and Lemma 54, just as is explained in the paper by Guruswami et al. [16, Theorem 4.7].

C Proof of Proposition 7: Extractors Are Good Samplers

Let $X \subseteq \{0,1\}^\ell$ be the set on the left-hand side of Equation (8). Since total variation distance is half ℓ_1 distance, for each $x \in X$,

$$\sum_{v \in V} |\Pr[f(U_s) = v] - \Pr[f(\mathsf{Ext}(x, U_d)) = v]| > \varepsilon |V|. \tag{110}$$

Therefore, by the triangle inequality, for each $x \in X$, there is some $v_x \in V$ such that

$$|\Pr[f(U_s) = v_x] - \Pr[f(\mathsf{Ext}(x, U_d)) = v_x]| > \varepsilon. \tag{111}$$

Partition $X = X_1 \cup \cdots \cup X_{|V|}$, where $\mathcal{X}_v = \{x \in X : v_x = v\}$. For each v, we can further partition X_v into $X_v^+ \cup X_v^-$, based on which term of the left hand side of Equation (111) is bigger.

Identify X_v^+ with a random variable that is uniformly distributed over the set X_v^+, and let $Y \sim U_d$ be independent of X_v^+. Then

$$\Pr[\mathsf{Ext}(X_v^+, Y) \in f^{-1}(v_x)] > \Pr[U_s \in f^{-1}(v_x)] + \varepsilon. \tag{112}$$

Therefore, by the extractor condition, $|X_v^+| \le 2^k$. Similarly, $|X_v^-| \le 2^k$, and hence $|X_v| \le 2^{k+1}$. By summing over all v, we conclude that $|X| \le 2^{k+1}|V|$ as claimed.

D Proof of Theorem 20: Derandomizing S-OW Branching Programs

The algorithm A of Theorem 20 is given in Algorithm 8. The analysis is similar to the proof of Theorem 8. The main difference is when we argue that the second hybrid distribution, H_2, simulates $\mathcal{P}$. (This argument has just two hybrid distributions.) Details follow.

Parameters

Just like in the proof of Theorem 8, we can assume without loss of generality that $T \le 2^S$. The block size h in Algorithm 8 is

$$h \stackrel{\text{def}}{=} \left\lfloor \frac{n}{3S^{c+1}} \right\rfloor. \tag{113}$$

Note that this time, the number of phases, r, is $\lceil T/h \rceil$, where h is the *block size*, in contrast to the proof of Theorem 8, where the number of phases was roughly T/B, where B is the *number of blocks*.

The algorithm A relies on Nisan's generator NisGen (Theorem 9). Naturally, the generator is instantiated with parameters S, T from the statement of Theorem 20. The error of NisGen is set at $\varepsilon \stackrel{\text{def}}{=} \frac{\exp(-cS)}{2r}$, just like in the proof of Theorem 8. Again, the seed length of NisGen is $s \le O(S \log T) \le O(S^2)$.

The algorithm A also relies on the Shaltiel-Umans extractor SUExt of Theorem 5. This extractor is instantiated with source length $\ell \stackrel{\text{def}}{=} n - 3h$, $\alpha \stackrel{\text{def}}{=} 1/2$, error

$$\varepsilon' \stackrel{\text{def}}{=} \frac{\exp(-cS)}{r \cdot 2^S}, \tag{114}$$

and entropy $k \stackrel{\text{def}}{=} \sqrt{n}$. This choice of k meets the hypotheses of Theorem 5, because $\log^{4/\alpha} \ell \le \log^8 n \le k$, and $S^{c+1} \le \sqrt{n}$, so $\log^{4/\alpha}(1/\varepsilon) \le \operatorname{polylog} n \le k$. Furthermore, by construction, $k^{1-\alpha} = n^{1/4} \ge s$ as long as $c \ge 4$ and n is sufficiently large, so we can think of SUExt_2 as outputting s bits.

Algorithm 8: The algorithm A of Theorem 20.

if $S^{c+1} > \sqrt{n}$ **then**
  Directly simulate $\mathcal{P}(v_0; x, U_T)$ using T random bits
else
  Partition $[n]$ into disjoint blocks, $[n] = I_1 \cup I_2 \cup \cdots \cup I_B$, where $|I_b| \approx h$. More precisely, let $B = \lceil n/h \rceil$, and let
  $I_b \leftarrow \{h \cdot (b-1) + 1, h \cdot (b-1) + 2, \ldots, \min\{h \cdot b, n\}\}$
  Let $I_0, I_{B+1} \leftarrow \varnothing$
  for $b \in [B]$ **do**
    Let $I'_b \leftarrow [n] \setminus (I_{b-1} \cup I_b \cup I_{b+1})$, with the largest elements removed so that $|I'_b| = n - 3h$
  end
  Initialize $v \leftarrow v_0$
  repeat r **times** `/* Here` $r \stackrel{\text{def}}{=} \lceil T/h \rceil$ `*/`
    Let $b \in [B]$ be such that $i(v) \in I_b$
    Let $I \leftarrow I'_b$
    Pick $y \in \{0,1\}^{O(S)}$ uniformly at random
    Update $v \leftarrow \mathcal{P}|_{[n] \setminus I}(v; x, \mathsf{NisGen}(\mathsf{SUExt}(x|_I, y)))$
  end
  return v
end

Efficiency

The runtime analysis of A is essentially the same as in the proof of Theorem 8; the only substantial difference is that the input to SUExt has length $\Theta(n)$, so SUExt takes $\text{poly}(n)$ time instead of $\text{poly}(S)$ time. Thus, overall, A runs in time $T \cdot \text{poly}(n, S)$. The space complexity and randomness complexity analyses are essentially the same as in the proof of Theorem 8.

Correctness

The proof of Equation (37) has the same structure as the proof of Equation (9). Assume without loss of generality that $S^{c+1} \leq \sqrt{n}$. The first hybrid distribution is defined by Algorithm 9. The number of "bad" inputs in Claim 55 is much lower than the number of "bad" inputs in Claim 11; intuitively, this is because A uses a *much larger* portion of the input as a source of randomness compared to the algorithm of Theorem 8.

▷ Claim 55 ($\mathsf{A} \approx \mathsf{H}_1$). Recall that ε' is the error of SUExt. Then

$$\#\{x \in \{0,1\}^n : \mathsf{A}(\mathcal{P}, v_0, x, T) \not\approx_{\varepsilon' r \cdot 2^{S-1}} \mathsf{H}_1(\mathcal{P}, v_0, x, T)\} \leq 2^{n/S^c}. \tag{115}$$

Proof sketch. The proof follows exactly the same reasoning as the proof of Claim 11. The number of bad x values is bounded by

$$\begin{aligned}
\#\text{ bad } x &\leq B \cdot 2^S \cdot 2^{n-|I'_b|} \cdot 2^{k+S+1} && (116)\\
&\leq 2^{3h+\sqrt{n}+O(S)} && (117)\\
&\leq 2^{n/S^{c+1}+\sqrt{n}+O(S)} && (118)\\
&\leq 2^{3n/S^{c+1}} && (119)\\
&\leq 2^{n/S^c} && (120)
\end{aligned}$$

for sufficiently large n. ◁

Algorithm 9: The algorithm H_1 defining the first hybrid distribution used to prove Equation (37). The only difference between A and H_1 is that H_1 picks a uniform random seed for NisGen, instead of extracting the seed from the input.

Initialize $v \leftarrow v_0$
repeat r **times**
  Let $b \in [B]$ be such that $i(v) \in I_b$
  Let $I \leftarrow I'_b$
  Pick $y' \in \{0,1\}^s$ uniformly at random
  Update $v \leftarrow \mathcal{P}|_{[n]\setminus I}(v; x, \mathsf{NisGen}(y'))$
end
return v

Algorithm 10: The algorithm H_2 defining the second hybrid distribution used to prove Equation (37). The only difference between H_1 and H_2 is that H_2 feeds true randomness to $\mathcal{P}|_{[n]\setminus I}$, instead of feeding it a pseudorandom string from Nisan's generator.

Initialize $v \leftarrow v_0$
repeat r **times**
  Let $b \in [B]$ be such that $i(v) \in I_b$
  Let $I \leftarrow I'_b$
  Pick $y'' \in \{0,1\}^T$ uniformly at random
  Update $v \leftarrow \mathcal{P}|_{[n]\setminus I}(v; x, y'')$
end
return v

The second hybrid distribution is defined by Algorithm 10.

▷ Claim 56 ($\mathsf{H}_1 \approx \mathsf{H}_2$). For every x,

$$\mathsf{H}_1(\mathcal{P}, v_0, x, T) \sim_{\varepsilon r} \mathsf{H}_2(\mathcal{P}, v_0, x, T), \tag{121}$$

where ε is the error of NisGen.

Proof sketch. The proof is the same as that of Claim 12. ◁

All that remains is the final step of the hybrid argument. In this case, H_2 actually simulates $\mathcal{P}$ with *no* error. This argument is where we finally use the fact that $\mathcal{P}$ only has sequential access to its input.

▷ Claim 57 ($\mathsf{H}_2 \sim \mathcal{P}$). For every x,

$$\mathsf{H}_2(\mathcal{P}, v_0, x, T) \sim \mathcal{P}(v_0; x, U_T). \tag{122}$$

Proof sketch. The set I'_b chosen by H_2 excludes every index in $[n]$ that is within h of $i(v)$. Therefore, each iteration of the loop in H_2 simulates at least h steps of $\mathcal{P}$. Since $r \geq T/h$, overall, H_2 simulates at least T steps of $\mathcal{P}$. But $T \geq \text{length}(\mathcal{P})$, so we are done, just like in the proof of Claim 14. ◁

Proof of Theorem 20. By Claims 55 to 57 and the triangle inequality,

$$\#\{x \in \{0,1\}^n : \mathsf{A}(\mathcal{P}, v_0, x, t) \not\sim_\delta \mathcal{P}(v_0; x, U_T)\} \leq 2^{n/S^c}, \tag{123}$$

where $\delta = \varepsilon r + \varepsilon' r \cdot 2^{S-1}$. By our choice of ε, the first term is at most $e^{-cS}/2$. By our choice of ε', the second term is also at most $e^{-cS}/2$. Therefore, $\delta \leq e^{-cS}$. ◀

E Proof of Theorem 49: Disambiguation with Advice

We begin with randomness-efficient amplification of Theorem 42; Lemma 58 is analogous to Lemma 36, and its proof follows the same reasoning. The details are included only for completeness.

▶ **Lemma 58.** *Fix $S : \mathbb{N} \to \mathbb{N}$ with $S(n) \geq \log n$ and $\delta : \mathbb{N} \to [0,1]$, both constructible in space $O(S)$. For every $L \in \mathbf{NSPACE}(S)$, there is a single-valued unambiguous algorithm $\mathcal{A}$ so that for every $x \in \{0,1\}^n$,*

$$\Pr_{y \in \{0,1\}^\infty}[\mathcal{A}(x,y) \in \{L(x), \perp_{\mathrm{r}}\}] = 1, \tag{124}$$

$$\Pr_{y \in \{0,1\}^\infty}[\mathcal{A}(x,y) = \perp_{\mathrm{r}}] \leq \delta(n). \tag{125}$$

Furthermore, $\mathcal{A}$ only reads the first $\log_2(1/\delta(n)) + O(S^2)$ bits of y and runs in space $O(S)$.

Proof. Let $\mathcal{R}$ be an $O(S)$-space reduction from L to STConn. For $x \in \{0,1\}^n$, $\mathcal{R}(x) \in \{0,1\}^{\overline{n}}$, where $\overline{n} = 2^{O(S)}$, and without loss of generality, $\overline{n}$ depends only on n. Let s be the number of random bits used by $\mathsf{vMPSeededAlg}$ on inputs of length $\overline{n}$, so that $s \leq O(\log^2 \overline{n}) = O(S^2)$.

Let $\ell = \lceil \log_2(1/\delta) \rceil + 2s + 2$, and let $\mathsf{GUVExt} : \{0,1\}^\ell \times \{0,1\}^d \to \{0,1\}^s$ be the $(2s, 0.1)$-extractor of Theorem 6, so that $d \leq O(\log\log(1/\delta) + \log S)$. On input $x \in \{0,1\}^n, y \in \{0,1\}^\ell$, run Algorithm 11.

Algorithm 11: The algorithm used to prove Lemma 58.

```
for z ∈ {0,1}^d do
    Let a ← vMPSeededAlg(R(x), GUVExt(y, z))
    if a ≠ ⊥_r then return a
end
return ⊥_r
```

Clearly, Algorithm 11 runs in space $O(S + d)$. Since δ is constructible in space $O(S)$, its denominator must have at most $2^{O(S)}$ digits. Therefore, $\delta \geq 2^{-2^{O(S)}}$ and $d \leq O(S)$, so the algorithm runs in space $O(S)$. Furthermore, it is clearly single-valued unambiguous, and it is "zero-error", i.e., Equation (124) holds. Finally, by Proposition 7, the number of y such that $\mathsf{vMPSeededAlg}(\mathcal{R}(x), \mathsf{GUVExt}(y,z)) = \perp_{\mathrm{r}}$ for every z is at most 2^{2s+2}, and hence the probability that the algorithm outputs $\perp_{\mathrm{r}}$ is at most $\frac{2^{2s+2}}{2^\ell} \leq \delta$. ◀

Proof of Theorem 49. Let $\mathcal{A}$ be the algorithm of Lemma 58 with $\delta < 2^{-n}/\varepsilon$. Let $m = m(n)$ be the number of random bits used by $\mathcal{A}$. Let $\mathcal{B}$ be the algorithm witnessing the fact that L is within ε of $\mathbf{USPACE}(S)$.

Given input $x \in \{0,1\}^n$ and advice $a \in \{0,1\}^m$, compute $a = \mathcal{A}(x,a)$. If $a \neq \perp_{\mathrm{r}}$, output a. If $a = \perp_{\mathrm{r}}$, output $\mathcal{B}(x)$. This algorithm clearly runs in $O(S)$ space, uses $n - \log_2(1/\varepsilon(n)) + O(S^2)$ bits of advice, and is unambiguous (in fact, single-valued unambiguous).

Now we argue that there is some advice string such that the algorithm succeeds on all inputs. Let $S \subseteq \{0,1\}^n$ be the set of inputs on which $\mathcal{B}$ fails. Consider picking an advice string a uniformly at random. For each string $x \in S$, $\Pr_a[\mathcal{A}(x,a) = \perp_{\mathrm{r}}] \leq \delta$. Therefore, by the union bound, the probability that there is some $x \in S$ such that $\mathcal{A}(x,a) = \perp_{\mathrm{r}}$ is at most $|S|\delta = \varepsilon \cdot 2^n \cdot \delta < 1$. Therefore, there is *some* advice string such that the algorithm succeeds on all inputs in S. Finally, for *any* advice string, the algorithm succeeds on all inputs in $\{0,1\}^n \setminus S$ by Equation (124). ◀

Optimal Short-Circuit Resilient Formulas

Mark Braverman
Department of Computer Science, Princeton University, USA
mbraverm@cs.princeton.edu

Klim Efremenko
Computer Science Department, Ben-Gurion University, Beer Sheba, Israel
klimefrem@gmail.com

Ran Gelles
Faculty of Engineering, Bar-Ilan University, Ramat Gan, Israel
ran.gelles@biu.ac.il

Michael A. Yitayew
Department of Computer Science, Princeton University, USA

Abstract

We consider fault-tolerant boolean formulas in which the output of a faulty gate is short-circuited to one of the gate's inputs. A recent result by Kalai et al. [FOCS 2012] converts any boolean formula into a resilient formula of polynomial size that works correctly if less than a fraction 1/6 of the gates (on every input-to-output path) are faulty. We improve the result of Kalai et al., and show how to efficiently fortify any boolean formula against a fraction 1/5 of short-circuit gates per path, with only a polynomial blowup in size. We additionally show that it is impossible to obtain formulas with higher resilience and sub-exponential growth in size.

Towards our results, we consider interactive coding schemes when noiseless feedback is present; these produce resilient boolean formulas via a Karchmer-Wigderson relation. We develop a coding scheme that resists up to a fraction 1/5 of corrupted transmissions *in each direction of the interactive channel*. We further show that such a level of noise is maximal for coding schemes with sub-exponential blowup in communication. Our coding scheme takes a surprising inspiration from Blockchain technology.

2012 ACM Subject Classification Theory of computation → Computational complexity and cryptography; Theory of computation → Interactive computation; Theory of computation → Design and analysis of algorithms

Keywords and phrases Circuit Complexity, Noise-Resilient Circuits, Interactive Coding, Coding Theory, Karchmer-Wigderson Games

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.10

Related Version A full version of the paper is available at `https://arxiv.org/abs/1807.05014`.

Funding *Mark Braverman*: Supported in part by an NSF CAREER award (CCF-1149888), NSF CCF-1525342, a Packard Fellowship in Science and Engineering, and the Simons Collaboration on Algorithms and Geometry.
Klim Efremenko: Supported in part by the Israel Science Foundation (ISF) through grant No. 1456/18.
Ran Gelles: Supported in part by the Israel Science Foundation (ISF) through grant No. 1078/17.

Acknowledgements The authors would like to thank Raghuvansh Saxena and the anonymous reviewer for spotting an error in a preliminary version of this manuscript.

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka; Article No. 10; pp. 10:1–10:22

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

1 Introduction

Kleitman, Leighton and Ma [21] asked the following question: assume you wish to build a logic circuit C from AND and OR gates, however, due to some confusion, some small amount of AND gates were placed in the box of the OR gates (and vice versa), and there is no way to distinguish between the two types of gates just by looking at them. Can you construct a "resilient" logic circuit C' that computes the same functionality of C, even if some (small amount) of the AND gates are replaced with OR gates (and vice versa)?

The above toy question is a special case of a more general type of noise (faulty gates) known as *short-circuit* noise. In this model, a faulty gate "short-circuits" one of its input-legs to the output-leg. That is, the output of the gate is determined by the value of one of its input-legs. The specific input that is connected to the output is determined by an all-powerful adversary, possibly as a function of the input to the circuit. This model is equivalent to a setting in which a faulty gate can be replaced with an arbitrary function g, as long as it holds that $g(0,0) = 0$ and $g(1,1) = 1$. Note that this type of noise is different from the so-called von Neumann noise model for circuits [29], in which the noise flips the value of each wire in the circuit independently with probability p. See [21, 19] and references therein for a comparison between these two separate models.

The first solution to the above question – constructing circuits that are resilient to short-circuit faults – was provided by Kleitman et al. [21]. They show that for any number e, a circuit of size $|C|$ gates can be transformed into a "resilient" circuit of size $|C'|$ that behaves correctly even if up to e of its gates are faulty (short-circuited), and it holds that $|C'| \leq O(e \cdot |C| + e^{\log 3})$.

Further progress was made by Kalai, Lewko, and Rao [19] showing, for any constant $\varepsilon > 0$, how to convert any formula[1] F of size $|F|$ into a resilient formula F' of size $|F'| = \text{poly}_\varepsilon(|F|)$ such that F' computes the same function that F computes, as long as at most $(\frac{1}{6} - \varepsilon)$-fraction of the gates in *any input-to-output path* in F' suffer from short-circuit noise. Kalai et al. explicitly leave open the question of finding the *optimal* fraction of faulty gates for a resilient formula F'.[2]

We make further progress on the above open question and show that $\frac{1}{5}$ is a tight bound on the tolerable fraction of faulty gates per input-to-output path, conditioned that the increase in the size of the formula is sub-exponential. Namely, we show how to convert any formula to a resilient version that tolerates up to a fraction $\frac{1}{5} - \varepsilon$ of short-circuit gates per path,

▶ **Theorem 1** (Main, informal). *For any $\varepsilon > 0$, any formula F can be efficiently converted into a formula F' of size $|F'| = \text{poly}_\varepsilon(|F|)$ that computes the same function as F even when up to $\frac{1}{5} - \varepsilon$ of the gates in any of its input-to-output paths are short-circuited.*

We also show that our bound is tight. Namely, for an arbitrary formula F, it is impossible to make a resilient version (of sub-exponential size in $|F|$) that tolerates a fraction $\frac{1}{5}$ (or more) of short-circuit gates per path.

▶ **Theorem 2** (Converse). *There exists a formula F for computing some function f, such that no formula F' of size $|F'| = o(\exp(|F|))$ that computes f is resilient to a fraction $\frac{1}{5}$ of short-circuit noise in any of its input-to-output paths.*

---

[1] A formula is a circuit in which each gate has fan-out 1.

[2] For instance, it is clear that if all the gates in an input-to-output path can be short-circuited (i.e., the fraction of noise is 1), then the adversary has full control on the output of the circuit. Hence, the optimal noise rate for formulas lies within the range $[\frac{1}{6}, 1]$.

Similar to the work of Kalai et al. [19], a major ingredient in our result is a transformation, known as the Karchmer-Wigderson transformation [20] (hereinafter, the KW-transformation), between a formula that computes a boolean function f, and a two-party interactive communication protocol for a task related to f which we denote the KW-game for f, or KW_f for short. Similarly, a reverse KW-transformation converts protocols back to formulas; see below for more details on the KW-transformation. The work of Kalai et al. adapts the KW-transformation to a noisy setting in which the formula may suffer from short-circuit noise, and the protocol may suffer from channel noise. The "attack plan" in [19] for making a given formula F resilient to short-circuit noise is (i) apply the KW-transformation to obtain an interactive protocol π; (ii) Convert π to a noise-resilient protocol π' that tolerates up to δ-fraction of noise; (iii) apply the (reverse) KW-transformation on π' to obtain a formula F'. The analysis of [19] shows that the obtained F' is resilient to $\delta/2$ fraction of noise in any of its input-to-output paths.

The interactive protocols π, π' are defined in a setting where the parties have access to a noiseless feedback channel – the sender learns whether or not its transmission arrived correctly at the other side. Building upon recent progress in the field of coding for interactive protocols (see, e.g., [11]), Kalai et al. [19] construct a coding scheme for interactive protocols (with noiseless feedback) that features resilience of $\delta = \frac{1}{3} - \varepsilon$ for any $\varepsilon > 0$; this gives their result. Note that a resilience of $\delta = 1/3$ is maximal for interactive protocols in that setting [8], suggesting that new techniques must be introduced in order to improve the result by [19].

The loss in resilience witnessed in step (iii) stems from the fact that short-circuit noise affects formulas in a "one-sided" manner: a short-circuit of an AND gate can only turn the output from 0 to 1. A short-circuit in an OR gate can only turn the output from 1 to 0. The noisy AND gates are thus decoupled from the noisy OR gates: if the output of the circuit is 0, any amount of short-circuited OR gates will keep the output 0 and if the output is 1 any amount of short-circuited AND gates will keep the output 1.

Informally speaking, this decoupling reduces by half the resilience of circuits generated by the KW-transformation. Assume the formula F' obtained from the above process is resilient to δ'-fraction of noise. Then F' is correct if on a specific input-to-output path (a) at most δ'-fraction of the AND gates are short-circuited, but also if (b) at most δ'-fraction of the OR gates are short-circuited. Since the noise is decoupled, from (a) and (b) we get that F outputs the correct value even when $2\delta'$-fraction of the gates on that input-to-output path are noisy. Yet, the resilience of F' originates in the resilience of π' (step (iii) above). The KW-transformation limits the resilience of F' by the resilience of π', i.e., $2\delta' \leq \delta$, leading to a factor 2 loss.

We revisit the above line of thought and take a more careful noise analysis. Instead of bounding the *total* fraction of noise by some δ, we consider the case where the noise from Alice to Bob is bounded by some α while the noise in the other direction is bounded by β. A similar approach used by Braverman and Efremenko [6], yields interactive protocols (without noiseless feedback) with maximal resilience. In more detail, assume that the protocol π communicates n symbols overall. We define an (α, β)-corruption as any noise that corrupts up to αn symbols sent by Alice and up to βn symbols sent by Bob. We emphasize that the noise *fraction* on Alice transmissions is higher than α, since Alice speaks less than n symbols overall; the global noise fraction in this case is $\alpha + \beta$.

This distinction may be delicate but is instrumental. The KW-transformation translates a protocol of length n that is resilient to (α, β)-corruptions into a formula which is resilient to up to αn short-circuited AND gates *in addition to* up to βn short-circuited OR gates. When $\alpha = \beta$ the obtained formula is resilient to up to α-fraction of short-circuited gates in any input-to-output path, avoiding the factor 2 loss in resilience.

1.1 Techniques overview

Achievability: Coding schemes for noisy channels with noiseless feedback

We obtain resilient formulas by employing the approach of [19] described above. In order to increase the noise resilience to its optimal level, we develop a novel coding scheme which is resilient to $(1/5-\varepsilon, 1/5-\varepsilon)$-corruptions, assuming noiseless feedback.

The mechanism of our coding scheme resembles, in a sense, the *Blockchain technology* [23]. Given a protocol π_0 that assumes reliable channels, the parties simulate π_0 message by message. These messages may arrive at the other side correctly or not, however, a noiseless feedback channel allows each party to learn which of its messages made it through. With this knowledge, the party tries to create a "chain" of correct messages. Each message contains a pointer to the last message that was not corrupted by the channel. As time goes by, the chain grows and grows, and indicates the entire correct communication of that party. An appealing feature of this mechanism is the fact that whenever a transmission arrives correctly at the other side, the receiver learns *all* the correct transmissions so far. On the other hand, the receiver never knows whether a single received transmission (and the chain linked to it) is indeed correct.

The adversarial noise may corrupt up to $(1/5-\varepsilon)n$ of the messages sent by each party. We think of the adversary as trying to construct a different, corrupt, chain. Due to its limited budget, at the end of the coding scheme one of two things may happen. Either it is the case that the correct chain is the longest, or it is the case where the longest chain contains in its prefix a sufficient amount of uncorrupted transmissions.

Indeed, if the adversary tries to create its own chain, its length is bounded by $(1/5-\varepsilon)n$, while the correct chain is of length $2n/5$ at the least.[3] On the other hand, the adversary can create a longer chain which forks off the correct chain. As a simple example, consider the case where a party sends $\approx 2n/5$ messages which go through uncorrupted. Now the adversary starts corrupting the transmissions and extends the correct chain with $(1/5-\varepsilon)n$ corrupt messages.[4] The corrupt forked chain is of length $2n/5+(1/5-\varepsilon)n$ and may be longer than the correct chain. However, in this case, the information contained in the *uncorrupted* prefix of the corrupt forked chain is sufficient to simulate the entire transcript of π_0.

Another essential part of our coding scheme is its ability to alter the order of speaking according to the observed noise.[5] Most previous work usually follows the succeeding intuition. If party's transmissions were corrupted, then the information contained in these transmissions still needs to reach the other side. Therefore, the coding scheme should allow that party to speak more times. In this work we take the opposite direction – the more a party is corrupted at the first part of the protocol, the *less* it speaks in the later part. The intuition here is that if the adversary has already wasted its budget on the party, it cannot corrupt much of the sequential transmissions and we can reduce their amount. A resembling approach appears in [1].

---

[3] The order of speaking in the coding scheme depends on the noise and is not alternating. Therefore, it is not necessary that a party speak half of the times. See discussion below.

[4] This attack assumes that there are $n/5$ additional rounds where the same party speak. This assumption is usually false and serves only for this intuitive (yet unrealistic) example.

[5] Protocols that change their length or order of speaking as a function of the observed noise are called *adaptive* [15, 1]. Since these decisions are noise-dependent, the parties may disagree on the identity of the speaker in each round, e.g., both parties may decide to speak in a given round, etc. We emphasize that due to the noiseless feedback there is always a consensus regarding whose turn it is to speak next. Hence, while our scheme has a non-predetermined order of speaking, the scheme is *non-adaptive* by the terminology of [8]; see discussion in [8] and in Section 6 of [11].

One hurdle we face in constructing our coding scheme comes from the need to communicate pointers to previous messages using a small (constant-size) alphabet. Towards this end, we first show a coding scheme that works with a large alphabet that is capable of pointing back to any previous transmission. Next, we employ a variable-length coding, replacing each pointer with a large number of messages over a constant-size alphabet. We prove that this coding does not harm the resilience, leading to a coding scheme with a constant-size alphabet and optimal resilience to $(1/5-\varepsilon, 1/5-\varepsilon)$-corruptions.

Converse: Impossibility Bound

The converse proof consists of two parts. First, we show that for certain functions, any protocol resilient to $(1/5, 1/5)$-corruptions must have an exponential blowup in the communication. In the second part, we show a (noisy) KW-transformation from formulas to protocols. Together, we obtain an upper bound on the noise of formulas. Indeed, assuming that there is a "shallow" formula that is resilient to $(1/5, 1/5)$-corruptions, converting it into a protocol yields a "short" protocol with resilience to $(1/5, 1/5)$-corruptions. The existence of such a protocol contradicts the bound of the first part.

The bound on the resilience of protocols follows a natural technique of confusing a party between two possible inputs. We demonstrate that a $(1/5, 1/5)$-corruption suffices in making one party (say, Alice) observe exactly the same transcript whether Bob holds y or y'. Choosing x, y, y' such that the output of the protocol differs between (x, y) and (x, y'), leads to Alice erring on at least one of the two instances.

This idea *does not* work if the protocol is allowed to communicate a lot of information. To illustrate this point, assume $f : \Sigma^n \times \Sigma^n \to \Sigma^z$ defined over a channel with alphabet Σ. Consider a protocol where the parties send their inputs to the other side encoded via a standard Shannon error-correcting code of length $n' = O(n)$ symbols, with distance $1-\varepsilon$ for some small constant $\varepsilon > 0$. The protocol communicates $2n'$ symbols overall, and a valid $(1/5, 1/5)$-corruption may corrupt up to $2n'/5$ symbols *of each one of the codewords*. However, this does not suffice to invalidate the decoding of either of the codewords, since an error correcting code with distance ≈ 1 is capable of correcting up to $\approx n'/2$ corrupted symbols.

On the other hand, once we limit the communication of the protocol, even moderately, to around n symbols, the above encoding is not applicable anymore. Quite informally, our lower bound follows the intuition described below. We show the existence of a function f such that for any protocol that computes f in r rounds (where r is restricted as mentioned above), the following properties hold for one of the parties (stated below, without loss of generality, for Alice). There are inputs x, x', y, y' such that (1) $f(x, y) \neq f(x', y) \neq f(x', y')$ and (2) Alice speaks at most $r/5$ times during the first $2r/5$ rounds. Further, (3) when Alice holds x, the protocol communicates exactly the same messages during its first $2r/5$ rounds, whether Bob holds y or y' (assuming no channel noise is present).

When we bound the protocol to these conditions, a $(1/5, 1/5)$-corruption is strong enough to make the transcript identical from Alice's point of view on (x', y) and (x', y'), implying the protocol cannot be resilient to such an attack. In more details, we now describe an attack and assume Bob speaks at most $2r/5$ times beyond round number $2r/5$, given the attack. [If Bob speaks more, then an equivalent attack will be able to confuse Bob rather than Alice.] The attack changes the first $2r/5$ rounds as if Alice holds x rather than x'; this amounts to corrupting at most $r/5$ transmissions by Alice due to property (2). Bob behaves the same regardless of its input due to property (3). From round $2r/5$ and beyond, the attack corrupts Bob's messages so that the next $r/5$ symbols Bob sends are consistent

with y and the following $r/5$ symbols Bob communicates are consistent with y'. Since Bob speaks less than $2r/5$ times (given the above noise), the attack corrupts at most $r/5$ of Bob's transmissions after round $2r/5$.

Unfortunately, while the above shows that some functions f cannot be computed in a resilient manner, this argument cannot be applied towards a lower bound on resilient formulas. The reason is that the KW_f task is not *a function*, but rather *a relation* – multiple outputs may be valid for a single input. The attack on protocols described earlier shows that a $(1/5, 1/5)$-corruption drives the protocol to produce a different output than in the noiseless instance. However, it is possible that a resilient protocol gives a different *but correct* output.

Therefore, we need to extend the above argument so it applies to computations of arbitrary relations. Specifically, we consider the parity function on n bits and its related KW-game. We show the existence of inputs that satisfy conditions (2) and (3) above while requiring that the outputs of different inputs be disjoint. I.e., any possible output of (x', y) is invalid for (x, y) and for (x', y').

The last part of the converse proof requires developing a KW-transformation from formulas to protocols, in a *noise-resilience preserving* manner. Let us begin with some background on the (standard) KW-transformation. The KW game (or rather a slight adaptation we need for our purposes) is as follows. For a boolean function f on $\{0,1\}^n$, Alice gets an input x such that $f(x) = 0$ and Bob gets an input y such that $f(y) = 1$, their goal is to output a literal function $\ell(z)$ (i.e. one of the $2n$ functions of the form $\ell(z) = z_i$ or $\ell(z) = \neg z_i$) such that $\ell(x) = 0$ and $\ell(y) = 1$.

Let F be a boolean formula for f, consisting of $\vee$ and $\wedge$ gates, and where all the negations are pushed to the input layer (i.e. F is a monotone formula of the literals z_i, $\neg z_i$). The conversion of F to a protocol π for the KW_f game is as follows. View the formula as the protocol tree, with the literals at the bottom of the tree being the output literal function. Assign each $\wedge$ node to Alice, and each $\vee$ node to Bob.

The invariant maintained throughout the execution of the protocol is that if the protocol reaches a node v, then the value of v in F is 0 when evaluated on x, and 1 when evaluated on y. Each time when the protocol is at node v and it is Alice's turn to speak (thus v is an $\wedge$ gate in F), Alice sends the identity of a child which evaluates to 0 on x. Note that assuming the invariant holds for v, Alice can send the identity of such a child (since one of the inputs to an AND gate which outputs a 0 also evaluates to 0), while this child must evaluate to 1 on y assuming v evaluates to 1 on y. By maintaining this invariant, Alice and Bob arrive at the bottom, where they reach a literal evaluating to 0 on x and 1 on y. Note that there is some room for arbitrary decision making: if more than one child of v evaluates to 0 on x, Alice is free to choose any such child – the protocol will be valid for any such choice.

In this work we extend the above standard KW-transformation to the noisy-regime. Namely, we wish to convert a *resilient* formula into an interactive protocol π *while keeping the protocol resilient* to a similar level of channel noise. We note that the extension we need is completely different from previous uses of the KW-transformation. Indeed, for the achievability bound, a KW-transformation is used both in steps (i) and (iii) in the above outline of [19]. However, the instance used in step (i) assumes there is no noise, while the instance in step (iii) works in the other direction, i.e., it transforms (resilient) protocols to (resilient) formulas.

Similar to the standard transformation, our noisy KW-transformation starts by constructing a protocol tree based on the formula's structure, where every $\wedge$-gate is assigned to Alice and any $\vee$-gate to Bob. The main difference is in the decision making of how to proceeds when reaching a node v. The goal is to keep the invariant that the gate v in F evaluates to 0 on x and to 1 on y, *even when noise is present.*

When only one of v's descendants evaluates to 0 on x in F, Alice has no choice but to choose that child. However, when more than a single descendant evaluates to 0 on x, Alice's decision is less obvious. Moreover, this decision may affect the resilience of the protocol – it is possible that noise causes one of the descendants evaluate to 1 on that given x.

We observe, however, that one of v's children evaluates to 0 on x given *all the noise patterns* F is resilient against. The other children may still evaluate to 1 sometimes, as a function of the specific noise. Once we identify this special child that always evaluates to 0, Alice can safely choose it and maintain the invariant (and the correctness of the protocol), regardless of future noise. Giving some more details, we prove that if such a special child did not exist and all descendants could evaluate to both 0 and 1 as a function of the noise, then we could construct a noise E^* that would make all descendants evaluate to 1 on x simultanously. Hence, assuming the noise is E^*, the node v would evaluate to 1 on x, and consequently $F(x) = 1$. At the same time, we show that F is resilient to the noise E^*, so $F(x) = 0$ assuming the noise is E^*, and we reach a contradiction.

1.2 Other related work

The field of interactive coding schemes [11] started with the seminal line of work by Schulman [26, 25, 27]. Commonly, the goal is to compile interactive protocols into a noise-resilient version that has (1) good noise resilience; (2) good rate; and (3) good probability of success. Computational efficiency is another desired goal. Numerous works achieve these goals, either fully or partially [5, 13, 3, 9, 6, 14, 22, 16, 10], where the exact parameters depend on the communication and noise model.

Most related to this work are coding schemes in the setting where a noiseless feedback channel is present. Pankratov [24] gave the first interactive coding scheme that assumes noiseless feedback. The scheme of [24] aims to maximize the rate of the scheme assuming all communication passes over a binary symmetric channel (BSC) with flipping parameter ε (i.e., a channel that communicates bits, where every bit is flipped with probability ε, independently of other bits). Pankratov's scheme achieves a rate of $1 - O(\sqrt{\varepsilon})$ when $\varepsilon \to 0$. Gelles and Haeupler [12] improved the rate in that setting to $1 - O(\varepsilon \log 1/\varepsilon)$, which is the current state of the art. For the regime of large noise, Efremenko, Gelles, and Haeupler [8] provided coding schemes with maximal noise resilience, assuming noiseless feedback. They showed that the maximal resilience depends on the channel's alphabet size and on whether or not the order of speaking is noise-dependent. Specifically, they developed coding schemes with a noise-independent order of speaking and a constant rate that are resilient to $1/4 - \varepsilon$ and $1/6 - \varepsilon$ fraction of noise with a ternary and binary alphabet, respectively. When the order of speaking may depend on the noise, the resilience increases to $1/3 - \varepsilon$ for any alphabet size. They show that these noise levels are optimal and that no general coding scheme can resist higher levels of noise.

There has been tremendous work on coding for noisy channels with noiseless feedback in the one-way (non-interactive) communication setting, starting with the work of Shannon, Horstein, and Berlekamp [28, 18, 2]. It is known that the presence of feedback does not change the channel's capacity, however, it improves the error exponent. The maximal noise-resilience in this setting is also known. Recently, Haeupler, Kamath, and Velingker [17] considered deterministic and randomized codes that assume a partial presence of feedback.

2 Preliminaries

Notations

For integers $i \leq j$ we denote by $[i, j]$ the set $\{i, i+1, \ldots, j\}$ any by $[i]$ the set $\{1, \ldots, i\}$. For a string $s \in \Sigma^*$ and two indices $x, y \in \{1, \ldots, |s|\}$, $x < y$ we let $s[x, y] = s_x s_{x+1} \cdots s_y$. We will treat $\emptyset$ as the empty word, i.e., for any $a \in \Sigma^*$ we have $a \circ \emptyset = \emptyset \circ a = a$. For bits $a, b \in \{0, 1\}$ we let $a \oplus b = a + b \mod 2$, and $\bar{b} = 1 - b$. All logarithms are taken to base 2, unless the base is explicitly written.

Interactive Protocols

In the interactive setting we have two parties, Alice and Bob, which receive private inputs $x \in X$ and $y \in Y$, respectively. Their goal is to compute some predefined function $f(x, y) : X \times Y \to Z$ by sending messages to each other. A *Protocol* describes for each party the next message to send, given its input and the communication received so far. We assume the parties send symbols from a fixed alphabet Σ. The protocol also determines when the communication ends and the output value (as a function of the input and received communication).

Formally, an interactive protocol π can be seen as a $|\Sigma|$-ary tree (also referred to as the *protocol tree*), where each node v is assigned either to Alice or to Bob. For any v node assigned to Alice there exists a mapping $a_v : X \to \Sigma$ that maps the next symbol Alice should send, given her input. Similarly, for each one of Bob's nodes we set a mapping $b_v : Y \to \Sigma$. Each leaf is labeled with an element of Z. The output of the protocol on input (x, y) is the element at the leaf reached by starting at the root node, and traversing down the tree where at each internal node v owned by Alice (resp., Bob), if $a_v(x) = i$ (resp., $b_v(y) = i$) the protocol advances to the i-th child of v. We conveniently denote Alice's nodes by the set V_a and Bob's nodes by the set V_b. We may assume that all the nodes in a given protocol tree are reachable by some input $(x, y) \in X \times Y$ (otherwise, we can prune that branch without affecting the behaviour of the protocol). Note that the order of speaking in π is not necessarily alternating and it is possible the same party is the sender in consecutive rounds. For any given transcript T, we denote $\pi(\cdot \mid T)$ the instance of π assuming the history T. Specifically, assuming Alice is the sender in the next round (assuming the history so far is T), then the next communicated symbol is $\pi(x \mid T)$.

The length of a protocol, denoted $|\pi|$, is the length of the longest root-to-leaf path in the protocol tree, or equivalently, it is the maximal number of symbols the protocol communicates in any possible instantiation. In the following we assume that all instances have the same length $|\pi|$. The communication complexity of the protocol is

$$CC(\pi) = |\pi| \log |\Sigma|.$$

When Σ is constant (independent of the input size), we have $CC(\pi) = O(|\pi|)$.

Transmission Noise with Feedback

We will assume the communication channel may be noisy, that is, the received symbol may mismatch with the sent symbol. All the protocols considered in this work assume the setting of *noiseless feedback*: the sender always learns the symbol that the other side received (whether corrupted or not). The receiver, however, does not know whether the symbol it received is indeed the one sent to him.

A noise pattern is defined as $E \in \{0, 1, \ldots, |\Sigma| - 1, *\}^{|V_a| \cup |V_b|}$. For any node v, E_v denotes the symbol that the receiver gets for the transmission that is done when the protocol reaches the node v. Specifically, say v is an Alice-owned node, then if $E_v = *$, Bob receives the symbol sent by Alice; otherwise, $E_v \neq *$, Bob receives the symbol E_v. Note that due to the feedback, Alice learns that her transmission was corrupted as well as the symbol that Bob received, and the protocol descends to the node dictated by E_v. We denote by π_E the protocol π when the noise is dictated by E; we sometimes write π_0 for a run of the protocol with no transmission noise, i.e., with the pattern $E = *^{|V_a| \cup |V_b|}$.

We say that a protocol is *resilient* to a noise pattern E if for any $(x, y) \in X \times Y$ it holds that π_E outputs the same value as π_0. While it is common to limit the noise to a constant fraction of the transmissions, in this work we take a more careful look at the noise, and consider the exact way it affects the transmissions of each party.

▶ **Definition 3.** *An (α, β)-corruption, is a noise pattern that changes at most $\alpha|\pi|$ symbols sent by Alice and at most $\beta|\pi|$ symbols sent by Bob. Note that the effective (combined) noise rate is $(\alpha + \beta)$.*

3 Resilience to $(1/5, 1/5)$-Corruptions is Impossible

In this section we prove that no coding scheme with constant overhead can be resilient to a $(1/5, 1/5)$-corruption. To this end we show a specific $(1/5, 1/5)$-corruption that confuses any protocol for a specific function f that is "hard" to compute in linear communication. Our results *does not* apply to coding schemes with vanishing rates. In fact, if the communication is exponentially large, coding schemes with resilience higher than $1/5$ exist.[6]

Normally, we discuss the case where protocols compute a *function* $f : X \times Y \to Z$. While our converse bound on the resilience of interactive protocols works for some hard function (e.g., the pointer jumping), such a proof does not suffice towards our converse on the resilience of boolean circuits (Theorem 2). The reason is that the conversion between formulas to protocols does not yield a protocol that computes a function but rather a protocol that computes a *relation*. Recall that for any given function f and any input (x, y) such that $f(x) = 0$ and $f(y) = 1$, the KW-game for f, KW_f, outputs an index $i \in [n]$ for which $x_i \neq y_i$ (see Section 5.1 for a formal definition). However, multiple such indices may exist and each such index is a valid output.

Let X, Y, Z be finite set and $R \subseteq X \times Y \times Z$ be a ternary relation. For any $(x, y) \in X \times Y$ and a given relation R let $R(x, y) = \{z \mid (x, y, z) \in R\}$ be the set of all z that satisfy the relation for x, y. Given such a relation, a protocol that *computes the relation* is the following two-party task. Alice is given $x \in X$ and Bob is given $y \in Y$. The parties need to agree on some $z \in R(x, y)$. We say that (x, y) is a *valid input* for R, if $x \in X$ and $y \in Y$. We assume that for any valid input, $|R(x, y)| > 0$.

We now show an explicit relation for which no protocol (of "short"' length) is resilient to $(1/5, 1/5)$ corruptions. Specifically, in the rest of this section we consider the binary parity function on n bits, $par : \{0, 1\}^n \to \{0, 1\}$, defined for any $x \in \{0, 1\}^n$ by

$$par(x) = x_1 \oplus \cdots \oplus x_n.$$

[6] For instance, consider the scheme in which each party sends its input to the other side encoded via a standard (Shannon) error-correcting code with distance ≈ 1. This trivial protocol is resilient to $(1/4 - \varepsilon, 1/4 - \varepsilon)$-corruption, yet its rate is 0.

Let $X = \{x \in \{0,1\}^n \mid par(x) = 0\}$ and $Y = \{y \in \{0,1\}^n \mid par(y) = 1\}$. We let $KW_{par} \subseteq X \times Y \times [n]$ be the KW-game for the parity function, defined by

$$KW_{par} = \{(x, y, z) \mid par(x) = 0 \wedge par(y) = 1 \wedge x_z \neq y_z\}.$$

We will need the following technical claim.

▷ Claim 4. Let $Y \subseteq \{0,1\}^n$. If $|Y| \geq 2^{n/2} + 1$ then there exist two elements $y_1, y_2 \in Y$ such that $\langle y_1, y_2 \rangle = 1$. Furthermore, if $|Y| \geq 2^{(n+1)/2} + 1$ then there exist two elements $y_1, y_2 \in Y$ such that $\langle y_1, y_2 \rangle = 0$.

Proof. Consider the linear space $L = \text{span}\{Y\}$. By a counting argument, it holds that $\dim(L) > n/2$. Let $L^\perp$ be the space orthogonal to L. Then $\dim(L) + \dim(L^\perp) = n$ and $\dim(L^\perp) < n/2$. Hence, $|L^\perp| < 2^{n/2}$ and thus $Y \not\subset L^\perp$. Let $y_1 \in Y \setminus L^\perp$. Since $y_1 \notin L^\perp$ it means that there must exist some element in Y, say y_2, for which $\langle y_1, y_2 \rangle \neq 0$. It follows that these two elements satisfy the claim, $\langle y_1, y_2 \rangle = 1$.

For the second part of the claim, let us construct $\tilde{Y} = \{(y, 1) \in \{0,1\}^{n+1} : y \in Y\}$; this is merely the set Y with an additional coordinate which is always set to one (over a space of dimension $n + 1$). Note that $|Y'| = |Y| \geq 2^{(n+1)/2} + 1$ and we can use the first part of this claim to show that there exist two elements $(y_1, 1), (y_2, 1) \in \tilde{Y}$ such that $\langle (y_1, 1), (y_2, 1) \rangle = 1$, therefore, $\langle y_1, y_2 \rangle = 0$. ◁

▶ **Lemma 5.** *Let π be an interactive protocol for KW_{par} (with inputs of n bits) of length $|\pi| = r$ defined over a communication channel with alphabet Σ and noiseless feedback. Without loss of generality, let Alice be the party who speaks less in the first $2r/5$ rounds of π. Additionally, assume $n/3 > 2r \log |\Sigma|/5 + 1$.*

Then, there exist inputs $x, x' \in X$, $y, y' \in Y$ for which:

(1) *$\pi(x, y)$ and $\pi(x, y')$ agree on the first $2r/5$ rounds.*

(2) *During the first $2r/5$ rounds of the execution $\pi(x, y)$ Alice speaks fewer times than Bob.*

(3) *$KW_{par}(x', y) \cap KW_{par}(x', y') = \emptyset$ and $KW_{par}(x', y) \cap KW_{par}(x, y) = \emptyset$.*

Note that the above lemma assumes Alice is the party that speaks fewer times in the first $2r/5$ rounds of π when averaging on all possible inputs $(x, y) \in X \times Y$; otherwise, a symmetric lemma holds for Bob.

Proof. Let x be an input for Alice such that on most y's Alice speaks fewer times in the first $2n/5$ rounds of $\pi(x, y)$. Such an input must exist by our choice of Alice. Let

$$Y' = \left\{ y \in Y \;\middle|\; CC_A^{\leq 2r/5}(\pi(x, y)) \leq CC_B^{\leq 2r/5}(\pi(x, y)) \right\}$$

be the set of all inputs for Bob, where Alice speaks fewer times in the first $2r/5$ rounds of π assuming Alice holds the above x. By the choice of x, it holds that $|Y'| \geq 2^n/2$.

Consider the set of transcript prefixes of length $2r/5$ generated by π when Alice holds the above x and Bob holds some input from the set Y',

$$T_x = \{t[1, 2r/5] \mid t = \pi(x, y), y \in Y'\}.$$

Note that there are at most $(2|\Sigma|)^{2r/5}$ different prefixes of length $2r/5$ over Σ with an arbitrary order of speaking. Since we assumed $n/3 > 2r \log |\Sigma|/5 + 1$, we have for large enough n,

$$|Y'| \geq 2^{n-1} \geq (2^{(n+1)/2} + 1)2^{n/3} \geq (2^{(n+1)/2} + 1)2^{2r \log |\Sigma|/5+1} \geq \Upsilon |T_x|,$$

with $\Upsilon = 2^{(n+1)/2} + 1$. Using a pigeon-hole principle, there must be $y^1, y^2, \ldots, y^\Upsilon \in Y'$ such that $\{\pi(x, y^i)\}_{i=1}^{\Upsilon}$ agree on the first $2r/5$ rounds of the protocol – they have an identical order of speaking and they communicate the same information.

Next consider the set $\{\overline{x} \oplus y^i\}_{i=1}^{\Upsilon}$. Claim 4 guarantees that there exist two elements in that set such that $\langle \overline{x} \oplus y^i, \overline{x} \oplus y^j \rangle = par(\overline{x})$; these y^i, y^j will be our y, y'.

Note that Properties (1) and (2) of the lemma are satisfied by the above x, y, y'. We are left to show an input x' for Alice that satisfies property (3).

Based on the above x, y, y' we construct x' in the following manner. For any $i \in [n]$ set

$$x'_i = \begin{cases} y_i & y_i = y'_i \\ \overline{x_i} & y_i \neq y'_i \end{cases}.$$

The above x' is constructed such that outputs given by KW_{par} are disjoint if we change only the input of Alice or only the input of Bob. Formally,

▷ Claim 6. The following claims hold for the above x, x', y, y'

a. $par(x') = 0$

b. $KW_{par}(x', y) \cap KW_{par}(x', y') = \emptyset$

c. $KW_{par}(x', y) \cap KW_{par}(x, y) = \emptyset$ and $KW_{par}(x', y') \cap KW_{par}(x, y') = \emptyset$

Proof.

a. It is easy to check that $x'_i = ((\overline{x_i} \oplus y_i) \cdot (\overline{x_i} \oplus y'_i)) \oplus \overline{x_i}$. Therefore,

$$\begin{aligned} par(x') &= \bigoplus_{i=1}^{n} x'_i = \bigoplus_{i=1}^{n} (((\overline{x_i} \oplus y_i) \cdot (\overline{x_i} \oplus y'_i)) \oplus \overline{x_i}) \\ &= \langle \overline{x} \oplus y, \overline{x} \oplus y' \rangle \oplus par(\overline{x}). \end{aligned}$$

Since we picked y, y' for which $\langle \overline{x} \oplus y, \overline{x} \oplus y' \rangle = par(\overline{x})$, we conclude that $par(x') = 0$.

b. Assume towards contradiction that $i \in KW_{par}(x', y) \cap KW_{par}(x', y')$, i.e., $x'_i \neq y_i$ as well as $x'_i \neq y'_i$. However, x'_i, y_i, y'_i are all bits and these two inequalities imply $y_i = y'_i$. But then, $x'_i = y_i$ by the way we construct x', which is a contradiction.

c. Assume towards contradiction that $i \in KW_{par}(x', y) \cap KW_{par}(x, y)$. That is, $x'_i \neq y_i$ and $x_i \neq y_i$ which means that $x'_i = x_i$. On the other hand, by the construction of x', either $x'_i \neq x_i$ or $x'_i = y_i$. Both options lead to a contradiction. The proof of the second part is identical.

◁

The first claim proves that x' is a valid input, i.e., $x' \in X$. The other claims prove property (3) of the lemma and conclude its proof. ◀

Our main result in this section is the following Theorem, proving that no protocol for the KW_{par} can be resilient to a $(1/5, 1/5)$-corruption if its communication is bounded. This will imply that any coding scheme that is resilient to $(1/5, 1/5)$-corruption must have rate 0. Specifically, it cannot produce a protocol with a constant overhead with respect to the optimal protocol that computes KW_{par} over reliable channels.

▶ **Theorem 7.** *Any interactive protocol π that computes the relation KW_{par} by at most $|\pi| < \frac{5}{6}\frac{n-3}{\log|\Sigma|}$ rounds over a noisy channel with alphabet Σ and noiseless feedback, is not resilient to $(1/5, 1/5)$-corruptions.*

Proof. Let π be a protocol with $r < \frac{5}{6}\frac{n-3}{\log|\Sigma|}$ rounds communicating symbols from the alphabet Σ. Via Lemma 5, let $x_0, x_1 \in X$ and $y_0, y_1 \in Y$ be inputs that satisfy

(1) $\pi(x_0, y_0)$ and $\pi(x_0, y_1)$ agree on the first $2r/5$ rounds
(2) During the first $2r/5$ bits of the protocol $\pi(x_0, y_0)$ Alice speaks less than Bob.
(3) $KW_{par}(x_1, y_0) \cap KW_{par}(x_1, y_1) = \emptyset$ and $KW_{par}(x_1, y_0) \cap KW_{par}(x_0, y_0) = \emptyset$

We now generate a simulated transcript T and show that T is consistent with a $(1/5, 1/5)$-corruption of $\pi(x_1, y_0)$. Additionally, it is either the case that T is consistent with a $(1/5, 1/5)$-corruption of $\pi(x_1, y_1)$ or it is consistent with a $(1/5, 1/5)$-corruption of $\pi(x_0, y_0)$. In the first case, Alice is unable to distinguish the case where Bob holds y_0 and y_1; in the second, Bob cannot tell if Alice holds x_0 or x_1. The outputs for different inputs are distinct by property (3). Thus the confused party is bound to err on at least one of them.

Note that the simulated transcript T contains messages *received* by the two parties, which may be noisy. Due to the feedback, both parties learn T. Additionally, the order of speaking in π is entirely determined by (prefixes of) T. Specifically, if two different instances of π have the same received transcript by round j, the party to speak in round $j + 1$ is identical in both instances.

The string T is obtained in the following manner:
1. Run $\pi(x_0, y_0)$ for $2r/5$ rounds. Let T_1 be the generated transcript.
2. Run $\pi(x_1, y_0 \mid T_1)$ until Bob transmits $r/5$ additional symbols (unless π terminates beforehand). Let T_2 be the generated transcript.
3. (if $|T_1T_2| < r$) Run $\pi(x_1, y_1 \mid T_1T_2)$ until Bob transmits $r/5$ additional symbols (unless π terminates beforehand).
4. (if $|T_1T_2T_3| < r$), let T_4 describe $\pi(x_1, y_0 \mid T_1T_2T_3)$ until it terminates.
5. Set $T = T_1T_2T_3T_4$.

In case the above algorithm didn't execute Step i, for $i \in \{3, 4\}$, assume $T_i = \emptyset$.

We now show that T corresponds to a $(1/5, 1/5)$-corrupted execution of π for two different valid inputs with disjoint outputs. We consider two cases: (i) when Step 3 halts since T reached its maximal size of r symbols (i.e., when $T_4 = \emptyset$), and (ii) when Step 3 halts since Bob transmitted $r/5$ symbols in this step ($T_4 \neq \emptyset$).

case (i) $T_4 = \emptyset$. In this case we show that a $(1/5, 1/5)$-corruption suffices to make the executions of $\pi(x_1, y_0)$ and $\pi(x_1, y_1)$ look the same from Alice's point of view.

Let Π be the transcript of a noisy execution of $\pi(x_1, y_0)$ (defined shortly) and split Π into three parts $\Pi = \Pi_1\Pi_2\Pi_3$ that correspond in length to T_1, T_2, T_3. The noise changes all Alice transmissions in Π_1 so that they correspond to Alice's symbols in T_1; the noise changes all Bob's transmissions in Π_3 so that they correspond to Bob's transmissions in T_3. It is easy to verify that the obtained transcript Π of received messages is exactly T. Furthermore, the first part changes at most $r/5$ transmissions by Alice, since by property (2) Alice speaks fewer times in the first $2r/5$ of the instance $\pi(x_0, y_0)$. The second part changes at most $r/5$ transmissions of Bob since T_3 halts before Bob communicates additional $r/5$ transmissions. Hence the noise described above is a valid $(1/5, 1/5)$-corruption.

On the other hand, and abusing notations, consider a (noisy) instance of $\pi(x_1, y_1)$ and let $\Pi = \Pi_1\Pi_2\Pi_3$ be the received messages transcript split to parts that corresponds in length to T_1, T_2, T_3, assuming the following noise. Again, the noise changes all Alice's transmissions in Π_1 to be the corresponding symbols received in T_1. This makes the $2r/5$ first rounds of the received transcript look as an instance $\pi(x_0, y_1)$. By Property (1), these transmissions agree with the first $2r/5$ transmissions in the noiseless instance $\pi(x_0, y_0)$; hence, the corrupted Π_1 equals T_1. Next, the noise changes Bob's transmissions in Π_2 to correspond to T_2. The obtained transcript Π is then exactly T. Again, T_1 contains at

most $2r/5$ of Alice's transmissions, and T_2 contains at most $r/5$ transmissions of Bob by their definition. Hence, this is a valid $(1/5, 1/5)$-corruption.

We conclude by recalling that $KW_{par}(x_1, y_0) \cap KW_{par}(x_1, y_1) = \emptyset$, then Alice must be wrong on at least one of the above executions, since her view in both executions is the same. Note that above proof holds even when $T_3 = \emptyset$.

case (ii) $T_4 \neq \emptyset$. In this case we show a $(1/5, 1/5)$-corruptions that makes the executions of $\pi(x_0, y_0)$ and $\pi(x_1, y_0)$ look the same from Bob's point of view. We point out that Alice speaks at most $r/5$ times after Step 1. Indeed, Step 1 contains $2r/5$ rounds, and Steps 2–3 contain $2r/5$ rounds where Bob speaks, hence Alice may speak in at most another $r/5$ times after Step 1.

Let Π be the transcript of a noisy execution of $\pi(x_0, y_0)$ where the noise is defined below. Split Π into 4 parts $\Pi = \Pi_1\Pi_2\Pi_3\Pi_4$ that correspond in length to T_1, T_2, T_3, T_4. The noise changes all Alice's transmissions in $\Pi_2\Pi_3\Pi_4$ so that they match the corresponding symbols of T_2, T_3, T_4. As mentioned, this corrupts at most $r/5$ symbols. Additionally, the noise changes Bob's transmissions in Π_3 to correspond to T_3; this by definition entails $r/5$ corruptions of Bob's transmissions. The obtained transcript Π is exactly T.

On the other hand, and abusing notations again, consider a noisy execution of $\pi(x_1, y_0)$ denoted by $\Pi = \Pi_1\Pi_2\Pi_3\Pi_4$. Here the noise is defined as follows. The noise changes all Alice's transmissions in Π_1 to match the corresponding symbols of T_1. As before, the noise changes Bob's transmissions in Π_3 to match T_3. Now it holds that $\Pi = T$, while the noise corrupted at most $r/5$ of each party's transmissions.

We conclude by recalling that $KW_{par}(x_0, y_0) \cap KW_{par}(x_1, y_0) = \emptyset$. Thus, Bob must be wrong on at least one of the above executions, since his view in both executions is exactly the same. ◀

Note that KW_{par} has a protocol of length $O(\log n)$ assuming reliable channels.[7] Theorem 7 leads to the following conclusion.

▶ **Corollary 8.** *There exists an interactive protocol π_0 defined over a noiseless channel with feedback such that any protocol π that computes the same functionality as π_0 and is resilient to $(1/5, 1/5)$-corruptions (assuming noiseless feedback) must incur with an exponential blowup in the communication.*

As a consequence, any coding scheme that compiles any protocol into a $(1/5, 1/5)$-resilient version, must have rate zero.

4 A Coding Scheme with Large Alphabet

In this section we construct a coding scheme for interactive protocols assuming a noiseless feedback. We show that for any constant $\varepsilon > 0$, any protocol π_0 defined over noiseless channels (with noiseless feedback) can be simulated by a protocol $\pi = \pi_\varepsilon$ defined over noisy channels (with noiseless feedback) such that (1) $CC(\pi)/CC(\pi_0) = O_\varepsilon(1)$, and (2) π is resilient to $(1/5 - \varepsilon, 1/5 - \varepsilon)$-corruptions. The protocol π in this section communicates symbols from a large alphabet of polynomial size in π_0. In the full version [7] we show how to reduce the size of the alphabet.

On a high level, the coding scheme π simulates π_0 step by step. The availability of a noiseless feedback channel allows a party to notice when the channel alters a transmission sent by that party. The next time that party speaks, it will re-transmit its message and

[7] This can easily be seen, e.g., by considering a formula that computes the parity of n bits, and applying the Karchmer-Wigderson transformation [20].

"link" the new transmission to its latest uncorrupted transmission. That is, each message carries a "link" – a pointer to a previous message sent by the same party. By following the links, the receiver learns the "chain" of uncorrupted transmissions; the party considers all "off-chain" transmissions as corrupted.

The Parse procedure (in Algorithm 1) parses all the transmissions received so far and outputs the "current chain": the (rounds of the) transmissions linked by the latest *received* transmission. Note that once a new transmission arrives, the current chain possibly changes. Moreover, upon reception of a corrupt transmission, a corrupt chain may be retrieved.

The TempTranscript procedure determines the partial simulated transcript of π_0 according to messages received in π so far, i.e., according to the current chains. Again, the scheme considers only transmissions that are on-chain and ignore all off-chain transmissions. The partial simulated transcript is defined as the concatenation of all the messages that (a) were received uncorrupted and (b) that were generated according to the correct information.

To clarify this issue, consider round i where, without loss of generality, Alice sends the message m_i. The latter property means that the last transmission *received* by Alice prior to round i, must be uncorrupted. This ensures that Alice learns which transmissions (so far) are correct and which are not, *in both directions*. It follows that Alice has full information about the on-going simulation of π_0. In particular, she can generate the correct m_i that extends the simulation of π_0 by one symbol. The former property ensures that m_i itself, the correct extension of the simulation of π_0, indeed arrives uncorrupted at the other side.

In each round of the protocol, the parties construct the partial transcript implied by messages received so far. If the received transmission is uncorrupt, the TempTranscript procedure retrieves the correct implied transcript (i.e., the implied transcript is indeed a prefix of the transcript of π_0). Then, the parties simulate the next rounds of π_0 assuming the implied partial transcript. As long as there is no noise in two alternating rounds, the next transmission extends the simulation of π_0 by one symbol. Otherwise, the sent symbol may be wrong, however, it will be ignored in future rounds once the chains indicate that this transmission was generated due to false information. Finally, at the end of the protocol, the parties output the transcript implied by the *longest* chain. The main part of this section is proving that the longest chain indeed implies a complete and correct simulation of π_0.

An important property of the coding scheme is its adaptive order of speaking. The first $2n/5$ rounds are alternating. On later rounds the order of speaking is determined according to observed noise: the more corrupted transmissions a party has, the *less* the party gets to speak. In particular, the protocol is split into *epochs* of 2 or 3 rounds each. In the first two rounds of an epoch, the order is fixed: Alice speaks in the first round and Bob speaks in the second. Then, the parties estimate the noise each party suffered so far (namely, the length of their current chain) and decide whether or not the epoch has a third round as well as who gets to speak in that extra round. For Alice to be the speaker in the third epoch-round, her *current* chain must be of length less than $n/5$ while Bob's current chain must be longer than $n/5$; Bob gets to speak if his chain is of length less than $n/5$ while Alice's chain is longer than $n/5$. In all other cases, the epoch contains only two rounds. We emphasize that due to the noiseless feedback, both parties agree on the received symbols (in both sides), which implies they agree on the current chains in both side, and thus, on the order of speaking in every epoch. The Next procedure, which determines the next speaker according to the current received transcript, captures the above idea.

The coding scheme is depicted in Algorithm 1. In the full version of this paper [7] we analyze Algorithm 1 and prove the following.

▶ **Theorem 9.** *For any $\varepsilon > 0$ and any binary alternating protocol π_0, Algorithm 1 correctly simulates π_0 over a noisy channel with noiseless feedback and is resilient to any $(1/5 - \varepsilon, 1/5 - \varepsilon)$-corruption.*

In the full version we also show how to reduce the channel's alphabet so it becomes independent of n, and depends only on the noise parameter ε. This is done via quite standard techniques of variable-length coding, cf. [5, 6, 4].

Algorithm 1 (**Part I**): A coding scheme against $(1/5, 1/5)$-corruptions assuming noiseless feedback (Large Alphabet; Alice's side).

Input: A binary alternating protocol π_0 with feedback; noise parameter $1/5 - \epsilon$. Alice's input for π_0 is x. Let $\Sigma = [n] \times \{0, 1, \emptyset\}$.

Throughout the protocol, maintain S_A, R_A, R_B, the sent, received by Alice, and received by Bob (as indicated by the feedback) symbols communicated up to the current round, respectively.
for $i = 1$ to $n = |\pi_0|/\varepsilon$ **do**
  $p_{\text{next}} = \textsc{Next}(R_A, R_B)$ ▷ Determine the next party to speak
  if p_{next} =Alice **then**
    $T \leftarrow \textsc{TempTranscript}(S_A, R_A, R_B)$
    The next symbol $\sigma = (link, b)$ to be communicated is:
      $link$ is the latest *non-corrupted* round $link < i$ where Alice is the speaker
        (0 if no such round exists).
      $b = \pi_0(x \mid T)$ if Alice is the sender in π_0, otherwise (or if π_0 has completed) $b = \emptyset$.
  else
    (receive a symbol from Bob)
  end if
end for

$j \leftarrow \arg\max |\mathsf{Parse}(R_B^{\le j})|$
$j' \leftarrow \arg\max |\mathsf{Parse}(R_A^{\le j'})|$
Output $\textsc{TempTranscript}(S_A, R_A^{\le j'}, R_B^{\le j})$

5 Applications for Circuits with Short-Circuit Noise

In this section we show that the KW-transformation between formulas and protocols (and vice versa) extends to the noisy setting in a manner that preserves noise-resilience. Applying the results of Section 4 onto the realm of boolean formulas gives a construction of formulas resilient to an optimal level of a fraction $(1/5 - \varepsilon)$ of short-circuit gates in any input-to-output path. Additionally, the results of Section 3 imply that noise-resilience of $1/5$ is maximal for formulas (assuming polynomial overhead).

In the following subsections, we show how to convert between formulas and protocols without affecting the noise-resilience. If we start with a formula that is resilient to (α, β)-corruptions, our transformation yields a protocol resilient to (α, β)-corruptions (Proposition 20). Moreover, given a protocol resilient to (α, β)-corruptions, the transformation yields a formula which is resilient to a similar level of noise (Proposition 23). Most proofs are deferred to the full version of this work.

Algorithm 1 (Part II): the Parse, Next, and TempTranscript Procedures.

procedure PARSE($m_1, \ldots, m_t$)
  $Chain \leftarrow \emptyset$; $j \leftarrow t$;
  while $j > 0$ **do**
    $Chain \leftarrow Chain \cup \{j\}$
    $j \leftarrow m_j.link$
  return $Chain$
end procedure

procedure TEMPTRANSCRIPT(S_A, R_A, R_B) ▷ Procedure for Alice; Bob's algorithm is symmetric
  Set $G_B = \text{PARSE}(R_A)$
  Set G_A as all the rounds in which outgoing transmissions are not corrupted (as learnt by R_B, S_A)
  For any $i < n$, if $\mathsf{Prev}(i), i \in G_A \cup G_B \cup \{0\}$ add i to *GoodChain*
  Set T as the concatenation of all $\{b_i\}_{i \in GoodChain}$, where $\sigma_i = (link_i, b_i)$ is the symbol received in round i.
  return T
end procedure

procedure NEXT(R_A, R_B)
  $i \leftarrow |R_A| + |R_B| + 1$ ▷ We are at round i
  $j \leftarrow 1$; $\mathsf{SkipCnt}_A \leftarrow 0$, $\mathsf{SkipCnt}_B \leftarrow 0$
  loop
    if $i = j$ **then return** Alice ▷ The speakers in the first two rounds of each epoch are fixed
    if $i = j + 1$ **then return** Bob
    ▷ Update the Skip counters
    if $|\text{PARSE}(R_A^{<j+2})| \leq n/5$ **then** $\mathsf{SkipCnt}_B \leftarrow \mathsf{SkipCnt}_B + 1$
    if $|\text{PARSE}(R_B^{<j+2})| \leq n/5$ **then** $\mathsf{SkipCnt}_A \leftarrow \mathsf{SkipCnt}_A + 1$

    if $|\mathsf{Parse}(R_B^{<j+2})| \leq n/5 < |\mathsf{Parse}(R_A^{<j+2})|$ **then** ▷ The epoch contains a 3rd round whenever one skip counter increases but the other does not
      if $i = j + 2$ **then return** Bob
      else $j \leftarrow j + 3$
    else if $|\mathsf{Parse}(R_A^{<j+2})| \leq n/5 < |\mathsf{Parse}(R_B^{<j+2})|$ **then**
      if $i = j + 2$ **then return** Alice
      else $j \leftarrow j + 3$
    otherwise
      $j \leftarrow j + 2$ ▷ An epoch with only 2 rounds
  end loop
end procedure

Note: $R_A^{\leq j}$ is the prefix of R_A as received by the j-th round of the protocol (incl. j) and $R_A^{<j}$ excluding round j. $R_B^{\leq j}$ and $R_B^{<j}$ are similarly defined.

5.1 Preliminaries

Formulas

A formula $F(z)$ over n-bit inputs $z \in \{0,1\}^n$ is a k-ary tree where each node is a $\{\wedge, \vee\}$ gate with fan-in k and fan-out 1. [While our results apply to any k, in this section we will usually assume $k = 2$ for simplicity.] Each leaf is a literal (either z_i or $\neg z_i$). The value of a node v given the input $z \in \{0,1\}$, denoted $v(z) \in \{0,1\}$, is computed in a recursive manner: the value of a leaf is the value of the literal (given the specific input z); the value of an $\wedge$ gate is the boolean AND of the values of its k descendants, $v_0, \cdots, v_{k-1}$, that is $v(z) = v_0(z) \wedge \cdots \wedge v_{k-1}(z)$. The value of an OR gate is $v(z) = v_0(z) \vee \cdots \vee v_{k-1}(z)$. The output of the formula on z, $F(z)$, is the value of the root node. We say that F computes the function $f : \{0,1\}^n \to \{0,1\}$ if for any $z \in \{0,1\}^n$ it holds that $F(z) = f(z)$.

The depth of a formula, denoted $\text{depth}(F)$, is the longest root-to-leaf path in it. The size of a formula, denoted $|F|$, is the number of nodes it contains. We denote by $V_\wedge$ the set of all the $\wedge$ nodes, and by $V_\vee$ the set of all the $\vee$ nodes.

Karchmer-Wigderson Games

For any boolean function $f : \{0,1\}^n \to \{0,1\}$, the *Karchmer-Wigderson game* is the following interactive task. Alice is given an input $x \in f^{-1}(0)$ and Bob gets $y \in f^{-1}(1)$. Their task is to find an index $i \in [n]$ such that $x_i \neq y_i$. We are guaranteed that such an index exists since $f(x) = 0$ while $f(y) = 1$. We denote the above task by KW_f.

Karchmer and Wigderson [20] proved the following relation between formulas and protocols.

▶ **Theorem 10** ([20]). *For any function $f : \{0,1\}^n \to \{0,1\}$, the depth of the optimal formula for f equals the length of the optimal interactive protocol for KW_f.*

▶ Remark 11. In the above, formulas are assumed to have fan-in 2 and protocols are assumed to communicate bits. However, the same reasoning and conversion applies also for a more general case, where each $\wedge$, $\vee$ gate has fan-in k, and the protocol sends symbols from alphabet of size $|\Sigma| = k$.

Furthermore, while our claims below are stated and proved assuming fan-in 2, all our claims apply to any arbitrary fan-in k.

Short-Circuit Noise

A short circuit noise replaces the value of a specific node with the value of one of its descendants. A noise pattern $E \in \{0, 1, \ldots, k-1, *\}^{|V_\wedge| \cup |V_\vee|}$ defines for each node whether it is short-circuited and to which input. Specifically, if for some node v, $E_v = *$ then the gate is not corrupted and it behaves as defined above. Otherwise, the value of the node is the value of its E_v-th descendant, $v(z) = v_{E_v}(z)$. We denote by F_E the formula with short circuit pattern E; we sometime write F for the formula with no short-circuit noise, i.e. with the noise pattern $E = *^{|V_\wedge| \cup |V_\vee|}$.

We say that a circuit is resilient to a noise pattern E if for any $z \in \{0,1\}^n$ it holds that $F(z) = F_E(z)$.

▶ **Definition 12.** *We say that F is resilient to δ-fraction of noise if it is resilient to all noise patterns E in which the fraction of corrupted gates in any input-to-output path in F is at most δ.*

We can also be more delicate and distinguish between noise in $\wedge$-gates and $\vee$-gates.

▶ **Definition 13.** *An (α, β)-corruption of short-circuit errors, is a noise pattern on a formula F of depth n that changes at most αn $\wedge$-gates and at most βn $\vee$-gates in any input-to-output path in F.*

The following is immediate by definition.

▷ Claim 14. If, for some $\delta > 0$, the formula F is resilient to any (δ, δ)-corruption of short-circuit errors, then F is also resilient to δ-fraction of noise.

On its surface, the other direction does not necessarily hold: (δ, δ)-corruption may corrupt up to a fraction 2δ of the gates in each path, hence resilience to δ-fraction appears to be insufficient to resist all (δ, δ)-corruptions. Nevertheless, we argue that these two notions are indeed equivalent. The reason is that a short-circuit in an $\wedge$-gate can only turn the output from 0 to 1. A short-circuit in an $\vee$-gate can only turn the output from 1 to 0. Then, if a formula evaluates to 1 on some input, the output remains 1 regardless of any amount of short-circuited $\wedge$-gates. If the output is 0, it remains so regardless of any number of short-circuited $\vee$-gates. This observation was already made by Kalai et al. [19].

▶ **Lemma 15** ([19, Claim 7])**.** *Let F be a formula, z an input and E any error pattern. Let $E_\wedge$ be the error pattern induced by E on the $\wedge$ gates alone (no errors in $\vee$ gates); Let $E_\vee$ be the error pattern induced by E on the $\vee$ gates alone. It holds that if $F_{E_\wedge}(z) = 0$, then $F_E(z) = 0$ and if $F_{E_\vee}(z) = 1$ then $F_E(z) = 1$.*

The above lemma then implies that resilience to δ-fraction of noise corresponds to resilience to the same fraction of noise in both type of gates.

▶ **Lemma 16.** *If, for some $\delta > 0$, the formula F is resilient to a fraction δ of short-circuit noise, then F is also resilient to any (δ, δ)-corruption.*

5.2 From Formulas to Protocols

We begin with a KW-transformation for noisy formulas, given a *specific* noise pattern.

▶ **Definition 17** (Noisy KW-transformation)**.** *For any formula $F(z)$ and any noise pattern E for F, the noisy transformation of F_E yields an interactive protocol π^{F_E} defined as follows over the domain $F_E^{-1}(0) \times F_E^{-1}(1)$.*

- *The formula-tree is converted into a protocol tree, where every $\wedge$ gate becomes a node where Alice speaks and every $\vee$ gate becomes a node where Bob speaks.*
- *For a node v, the mapping $a_v(z)$ for $z \in F_E^{-1}(0)$ and the mapping $b_v(z)$ for $z \in F_E^{-1}(1)$ are set as follows. Consider the evaluation of the formula F_E on z.*
  - *If v is an $\wedge$ gate, write $v(z) = v_0(z) \wedge v_1(z)$ where v_0 and v_1 are v's left and right descendants in F, respectively. For any $z \in F_E^{-1}(0)$, if $v_0(z) = 0$ we set $a_v(z) = 0$; otherwise we set $a_v(z) = 1$.*
  - *For an $\vee$ gate and $z \in F_E^{-1}(1)$ denote $v(z) = v_0(z) \vee v_1(z)$, and set $b_v(z) = 0$ if $v_0(z) = 1$; otherwise $b_v(z) = 1$.*
- *A leaf of F marked with the literal z_i or $\neg z_i$ becomes a leaf (output) of the protocol with the same literal.*

▶ Remark 18. In the above definition, we assume that if both $v_0(z) = 0$ and $v_1(z) = 0$ (for $z \in F^{-1}(0)$), the protocol continues to the left child. This choice is arbitrary, and any other choice is valid and gives an alternative protocol which still satisfies Proposition 19 below.

For instance, we can have non-intersecting sets Z_0 and Z_1 that determine the inputs z for which we take the left or right child, respectively (assuming both subformulas evaluate to 0 exactly on $Z_0 \cup Z_1$).

Following the mapping between formulas and protocols, [19] made the observations that a short-circuit error in a formula translates to channel noise in the equivalent KW protocol, assuming both parties learn the noise, i.e., assuming noiseless feedback. We will sometimes abuse notations and identify a short-circuit noise pattern with a transmission noise pattern for a formula F and a protocol π that share the same underlying tree structure. Furthermore, we will denote the two different objects with the same identifier E.

Next, we claim that performing the protocol π^{F_E} assuming the channel noise E computes the KW game of the noisy formula F_E.

▶ **Proposition 19.** *Assume that $F_E(z)$ computes the function $f(z)$. Then, $\pi_E^{F_E}$ computes KW_f.*

The proof goes by induction, similar to the original KW proof. The short-circuit noise forces some gate's output to be the output of a specific sub-formula. At the same time, channel noise causes the protocol to proceed to the corresponding sub-protocol. Together, we show that the following invariant holds, given the noise E: for any reachable node v, $v(x) = 0$ and $v(y) = 1$. This implies that once a leaf is reached, Alice and Bob disagree on its value, hence it is a valid output for the KW-game.

The above suggests that, for a given noise E, we can construct a protocol resilient to E. The next proposition proves that this can be extended to a family of noise patterns. This yields our main proposition for converting formulas to protocols in a noise-preserving way.

▶ **Proposition 20.** *Let F be a (complete) formula that computes the function f and is resilient to (α, β)-corruption of short-circuit gates in every input-to-output path. Then, a noisy KW-transformation yields a protocol π (over channels with feedback) that solves KW_f and is resilient to (α, β)-corruptions.*

The conversion from resilient formulas into resilient protocols of Proposition 20 implies an upper bound on the maximal resilience of formulas, and proves Theorem 2.

▶ **Theorem 21.** *There exists a function $f : \{0,1\}^n \to Z$ such that no formula F that computes f with fan-in k and depth less than $r < \frac{5}{6}\frac{n-3}{\log|k|}$, is resilient to a fraction $1/5$ of short-circuit noise.*

Proof. For $z \in \{0,1\}^n$, let $par(z) = z_1 \oplus \cdots \oplus z_n$ be the parity function.

Let F be a formula that computes $par(z)$ with AND/OR gates of fan-in k and $depth(F) < \frac{5}{6}\frac{n-3}{\log k}$. Assume that F is resilient to a fraction $1/5$ of short-circuit noise. Lemma 16 shows that F is also resilient to $(1/5, 1/5)$-corruptions of short-circuits. Moreover, assume that the formula's underlying graph is a complete k-ary tree.[8] Then, using Proposition 20 we obtain an interactive protocol π for KW_{par} of length $|\pi| = depth(F) < \frac{5}{6}\frac{n-3}{\log k}$ that communicates symbol from alphabet of size $|\Sigma| = k$, and is resilient to $(1/5, 1/5)$-corruptions. This contradicts Theorem 7. ◀

[8] If F has a node that has missing children, we can duplicate one of its children to obtain a complete graph. This clearly does not change the functionality of F, nor its resilience.

Note that computing the parity of n bits can be done with a formula of depth $O(\log n)$. However, the above theorem shows that any resilient formula for the parity function will have an exponential blow-up in depth, and thus exponential blow-up in size.

▶ **Corollary 22.** *There is no coding scheme that converts any formula F of size s into a formula F' of size $o(\exp(s))$, such that F' computes the same function as F and is resilient to $1/5$-fraction of short-circuit gates on every input to output path.*

5.3 From Protocols to Formulas

Here we would like to prove that a resilient protocol implies a resilient formula.

▶ **Proposition 23.** *Let π be a protocol that solves KW_f for some function f and is resilient to (α, β)-corruption. The KW-transformation on the reachable protocol tree of π yields a formula F that computes f and is resilient to (α, β)-corruption of short-circuit noise in any of its input-to-output paths.*

The above proposition is in fact a reformulation of a result by Kalai, Lewko, and Rao [19], implied by Lemma 15 and by Lemma 8 in [19].

Using our coding scheme that is resilient to $(1/5 - \varepsilon, 1/5 - \varepsilon)$-corruptions we get that we can fortify any formula F so it becomes resilient to $(1/5 - \varepsilon)$-fraction of short-circuit noise, with only polynomial growth in size.

▶ **Theorem 24.** *For any $\varepsilon > 0$, any formula F of depth n and fan-in 2 that computes a function f can be efficiently converted into a formulas F' that computes f even up to $1/5 - \varepsilon$ of the gates in any of its input-to-output path are short-circuited. F' has a constant fan-in $O_\varepsilon(1)$ and depth $O(n/\varepsilon)$.*

Theorem 1 is an immediate corollary of the above theorem, by noting that

$$|F'| \leq k^{\mathrm{depth}(F')} = \left(2^{O(\log(1/\varepsilon))}\right)^{O((\log|F|)/\varepsilon)} = \mathrm{poly}_\varepsilon(|F|).$$

Here $k \approx \varepsilon^{-2}$ is the fan-in of $|F'|$ given by the alphabet size of the resilient interactive protocol π' constructed earlier.

References

1. Shweta Agrawal, Ran Gelles, and Amit Sahai. Adaptive Protocols for Interactive Communication. In *2016 IEEE International Symposium on Information Theory (ISIT)*, pages 595–599, 2016. doi:10.1109/ISIT.2016.7541368.
2. Elwyn R. Berlekamp. *Block coding with noiseless feedback*. PhD thesis, Massachusetts Institute of Technology, 1964.
3. Zvika Brakerski, Yael T. Kalai, and Moni Naor. Fast Interactive Coding Against Adversarial Noise. *J. ACM*, 61(6):35:1–35:30, December 2014. doi:10.1145/2661628.
4. M. Braverman, R. Gelles, J. Mao, and R. Ostrovsky. Coding for Interactive Communication Correcting Insertions and Deletions. *IEEE Transactions on Information Theory*, 63(10):6256–6270, October 2017. doi:10.1109/TIT.2017.2734881.
5. M. Braverman and A. Rao. Toward Coding for Maximum Errors in Interactive Communication. *Information Theory, IEEE Transactions on*, 60(11):7248–7255, November 2014. doi:10.1109/TIT.2014.2353994.
6. Mark Braverman and Klim Efremenko. List and Unique Coding for Interactive Communication in the Presence of Adversarial Noise. *SIAM Journal on Computing*, 46(1):388–428, 2017. doi:10.1137/141002001.

7 Mark Braverman, Klim Efremenko, Ran Gelles, and Michael A. Yitayew. Optimal Short-Circuit Resilient Formulas. *CoRR*, arXiv:1807.05014, 2018. URL: http://arxiv.org/abs/1807.05014.

8 K. Efremenko, R. Gelles, and B. Haeupler. Maximal Noise in Interactive Communication Over Erasure Channels and Channels With Feedback. *IEEE Transactions on Information Theory*, 62(8):4575–4588, August 2016. doi:10.1109/TIT.2016.2582176.

9 Matthew Franklin, Ran Gelles, Rafail Ostrovsky, and Leonard J. Schulman. Optimal Coding for Streaming Authentication and Interactive Communication. *Information Theory, IEEE Transactions on*, 61(1):133–145, January 2015. doi:10.1109/TIT.2014.2367094.

10 R. Gelles, B. Haeupler, G. Kol, N. Ron-Zewi, and A. Wigderson. Explicit Capacity Approaching Coding for Interactive Communication. *IEEE Transactions on Information Theory*, 64(10):6546–6560, October 2018. doi:10.1109/TIT.2018.2829764.

11 Ran Gelles. Coding for Interactive Communication: A Survey. *Foundations and Trends® in Theoretical Computer Science*, 13(1–2):1–157, 2017. doi:10.1561/0400000079.

12 Ran Gelles and Bernhard Haeupler. Capacity of Interactive Communication over Erasure Channels and Channels with Feedback. *SIAM Journal on Computing*, 46(4):1449–1472, 2017. doi:10.1137/15M1052202.

13 Ran Gelles, Ankur Moitra, and Amit Sahai. Efficient Coding for Interactive Communication. *Information Theory, IEEE Transactions on*, 60(3):1899–1913, March 2014. doi:10.1109/TIT.2013.2294186.

14 Mohsen Ghaffari and Bernhard Haeupler. Optimal Error Rates for Interactive Coding II: Efficiency and List Decoding. In *Proceedings of the IEEE Symposium on Foundations of Computer Science*, FOCS '14, pages 394–403, 2014. doi:10.1109/FOCS.2014.49.

15 Mohsen Ghaffari, Bernhard Haeupler, and Madhu Sudan. Optimal Error Rates for Interactive Coding I: Adaptivity and Other Settings. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, STOC '14, pages 794–803, 2014. doi:10.1145/2591796.2591872.

16 Bernhard Haeupler. Interactive Channel Capacity Revisited. In *Proceedings of the IEEE Symposium on Foundations of Computer Science*, FOCS '14, pages 226–235, 2014. doi:10.1109/FOCS.2014.32.

17 Bernhard Haeupler, Pritish Kamath, and Ameya Velingker. Communication with Partial Noiseless Feedback. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2015)*, volume 40 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 881–897. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2015. doi:10.4230/LIPIcs.APPROX-RANDOM.2015.881.

18 M. Horstein. Sequential transmission using noiseless feedback. *IEEE Transactions on Information Theory*, 9(3):136–143, July 1963. doi:10.1109/TIT.1963.1057832.

19 Y. T. Kalai, A. Lewko, and A. Rao. Formulas Resilient to Short-Circuit Errors. In *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*, pages 490–499, October 2012. doi:10.1109/FOCS.2012.69.

20 Mauricio Karchmer and Avi Wigderson. Monotone Circuits for Connectivity Require Super-Logarithmic Depth. *SIAM Journal on Discrete Mathematics*, 3(2):255–265, 1990. doi:10.1137/0403021.

21 Dan Kleitman, Tom Leighton, and Yuan Ma. On the Design of Reliable Boolean Circuits That Contain Partially Unreliable Gates. *J. Comput. Syst. Sci.*, 55(3):385–401, December 1997. doi:10.1006/jcss.1997.1531.

22 Gillat Kol and Ran Raz. Interactive channel capacity. In *STOC '13: Proceedings of the 45th annual ACM Symposium on theory of computing*, pages 715–724, 2013. doi:10.1145/2488608.2488699.

23 Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008. URL: https://bitcoin.org/bitcoin.pdf.

24 Denis Pankratov. On the Power of Feedback in Interactive Channels. [Online:] http://people.cs.uchicago.edu/~pankratov/papers/feedback.pdf, 2013.

25 Sridhar Rajagopalan and Leonard Schulman. A coding theorem for distributed computation. In *STOC '94: Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*, pages 790–799, 1994. doi:10.1145/195058.195462.

26 Leonard J. Schulman. Communication on noisy channels: a coding theorem for computation. *Foundations of Computer Science, Annual IEEE Symposium on*, pages 724–733, 1992. doi:10.1109/SFCS.1992.267778.

27 Leonard J. Schulman. Coding for interactive communication. *IEEE Transactions on Information Theory*, 42(6):1745–1756, 1996. doi:10.1109/18.556671.

28 C. Shannon. The zero error capacity of a noisy channel. *IRE Transactions on Information Theory*, 2(3):8–19, September 1956. doi:10.1109/TIT.1956.1056798.

29 John von Neumann. Probabilistic logics and the synthesis of reliable organisms from unreliable components. In C. E. Shannon and J. McCarthy, editors, *Automata studies*, volume 34 of *Annals of Mathematics Studies*, pages 43–98. Princeton University Press, Princeton, 1956.

A Time-Distance Trade-Off for GDD with Preprocessing – Instantiating the DLW Heuristic

Noah Stephens-Davidowitz
Massachusetts Institute of Technology, Cambridge, MA, USA
http://www.noahsd.com
noahsd@gmail.com

Abstract

For $0 \leq \alpha \leq 1/2$, we show an algorithm that does the following. Given appropriate preprocessing $P(\mathcal{L})$ consisting of $N_\alpha := 2^{O(n^{1-2\alpha}+\log n)}$ vectors in some lattice $\mathcal{L} \subset \mathbb{R}^n$ and a target vector $\boldsymbol{t} \in \mathbb{R}^n$, the algorithm finds $\boldsymbol{y} \in \mathcal{L}$ such that $\|\boldsymbol{y} - \boldsymbol{t}\| \leq n^{1/2+\alpha}\eta(\mathcal{L})$ in time $\mathrm{poly}(n) \cdot N_\alpha$, where $\eta(\mathcal{L})$ is the smoothing parameter of the lattice.

The algorithm itself is very simple and was originally studied by Doulgerakis, Laarhoven, and de Weger (to appear in PQCrypto, 2019), who proved its correctness under certain reasonable heuristic assumptions on the preprocessing $P(\mathcal{L})$ and target $\boldsymbol{t}$. Our primary contribution is a choice of preprocessing that allows us to prove correctness without any heuristic assumptions.

Our main motivation for studying this is the recent breakthrough algorithm for IdealSVP due to Hanrot, Pellet–Mary, and Stehlé (to appear in Eurocrypt, 2019), which uses the DLW algorithm as a key subprocedure. In particular, our result implies that the HPS IdealSVP algorithm can be made to work with fewer heuristic assumptions.

Our only technical tool is the discrete Gaussian distribution over $\mathcal{L}$, and in particular, a lemma showing that the one-dimensional projections of this distribution behave very similarly to the continuous Gaussian. This lemma might be of independent interest.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms

Keywords and phrases Lattices, guaranteed distance decoding, GDD, GDDP

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.11

Related Version A full version of the paper is available at http://arxiv.org/abs/1902.08340. (Please read the full version!)

Acknowledgements I thank Guillaume Hanrot, Thijs Laarhoven, Alice Pellet–Mary, Oded Regev, and Damien Stehlé for helpful discussions. I also thank Alice Pellet–Mary, Guillaume Hanrot, and Damien Stehlé for sharing early versions of their work with me. I am also grateful to the CCC 2019 reviewers for their very helpful comments, and Daniel Dadush for showing me how to obtain the stronger results to be written up in the full version.

1 Introduction

A lattice $\mathcal{L} \subset \mathbb{R}^n$ is the set of all integer linear combinations

$$\mathcal{L} := \{z_1\boldsymbol{b}_1 + \cdots + z_n\boldsymbol{b}_n \ : \ z_i \in \mathbb{Z}\}$$

of linearly independent basis vectors $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n \in \mathbb{R}^n$. For a lattice $\mathcal{L} \subset \mathbb{R}^n$ and target vector $\boldsymbol{t} \in \mathbb{R}^n$, the d-Guaranteed Distance Decoding problem (d-GDD, or just GDD) asks us to find $\boldsymbol{y} \in \mathcal{L}$ such that $\|\boldsymbol{y} - \boldsymbol{t}\| \leq d$ for some distance $d := d(\mathcal{L})$ that depends only on $\mathcal{L}$. In particular, we must have $d \geq \mu(\mathcal{L})$, where $\mu(\mathcal{L}) := \max \mathrm{dist}(\boldsymbol{t}, \mathcal{L})$ is the covering radius of the lattice.

GDD with preprocessing (GDDP) is the variant of this problem in which we are allowed to perform arbitrary preprocessing on the lattice (but not on $\boldsymbol{t}$). I.e., formally an "algorithm" for GDDP is really a *pair* of algorithms, a preprocessing algorithm, which takes as input (a

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka; Article No. 11; pp. 11:1–11:8
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

basis for) a lattice $\mathcal{L} \subset \mathbb{R}^n$ and outputs some preprocessing $P(\mathcal{L})$, and a query algorithm which takes as input $P(\mathcal{L})$ and a target $\boldsymbol{t}$ and outputs a valid solution to the GDD instance $(\mathcal{L}, \boldsymbol{t})$. The complexity measure that interests us for such algorithms is the running time of the query algorithm.

In [7], Doulgerakis, Laarhoven, and de Weger (DLW) gave an elegant algorithm for GDDP whose correctness relies on certain heuristic assumptions. (Our presentation here differs quite a bit from DLW's. See Section 1.2.) In fact, [7] gave a family of algorithms parameterized by $0 \leq \alpha \leq 1/2$ whose preprocessing consists of $N_\alpha \approx 2^{n^{1-2\alpha}}$ lattice vectors in $\mathcal{L}$ whose length is roughly r. Given a target $\boldsymbol{t}$, the query algorithm starts by setting $\boldsymbol{t}' = \boldsymbol{t}$. The algorithm then simply searches for a vector $\boldsymbol{y}$ in the preprocessing list and an integer k such that $\|k\boldsymbol{y} - \boldsymbol{t}'\| < \|\boldsymbol{t}'\|$. If it finds one, it replaces $\boldsymbol{t}'$ by $\boldsymbol{t}' - k\boldsymbol{y}$ and repeats this procedure. Finally, it outputs $\boldsymbol{y}' := \boldsymbol{t} - \boldsymbol{t}' \in \mathcal{L}$. Under certain heuristic assumptions that in particular imply that the preprocessing is nicely distributed, [7] showed that this algorithm terminates with $\|\boldsymbol{y}' - \boldsymbol{t}\| = \|\boldsymbol{t}'\| \lesssim n^\alpha \cdot r$ in time $\mathrm{poly}(n) \cdot N_\alpha$.

DLW's algorithm is the first to provide a smooth trade-off between the running time and the distance d. (Such trade-offs are known for other lattice problems. E.g., without preprocessing, block reduction [19, 8] algorithms accomplish this for many lattice problems, and with preprocessing, such trade-offs are known for Bounded Distance Decoding and the Closest Vector Problem [12, 6].) This recently found an exciting application discovered by Pellet–Mary, Hanrot, and Stehlé [18]. [18] showed the best known time-approximation-factor trade-off for the very important problem of finding short non-zero vectors in ideal lattices (given suitable preprocessing on the underlying number field). Their algorithm uses the DLW algorithm as a key subprocedure. However, since DLW's algorithm relies on certain heuristic assumptions, their application crucially relies on the (reasonable but unproven) assumption that these heuristics apply in their particular use case.

1.1 Removing the heuristic in DLW's GDDP algorithm

We show how to instantiate DLW's heuristic algorithm in a provably correct way. In particular, we show an explicit distribution over the lattice such that, when the preprocessing consists of independent samples from this distribution, the above algorithm provably succeeds with high probability. Indeed, there is a very natural choice for this distribution: the discrete Gaussian over the lattice, $D_{\mathcal{L},s}$. This is the distribution that assigns probability to each lattice vector $\boldsymbol{y} \in \mathcal{L}$ proportional to its Gaussian mass $\exp(-\pi\|\boldsymbol{y}\|^2/s^2)$, and it is a ubiquitous tool in lattice algorithms and the study of lattices more generally. (See, e.g., [20].) When the width parameter $s > 0$ is at least as large as the *smoothing parameter* $\eta(\mathcal{L})$, the discrete Gaussian distribution $D_{\mathcal{L},s}$ provably behaves quite similarly to the continuous Gaussian in many ways [15]. (E.g., its moments are close to those of a continuous Gaussian.) So, one might expect that it will be distributed nicely enough to work for DLW's use case.

We show that for $s = \eta(\mathcal{L})$, the discrete Gaussian $D_{\mathcal{L},s}$ does in fact suffice to provably instantiate DLW's heuristic algorithm with $r \approx \sqrt{n} \cdot \eta(\mathcal{L})$. (This is essentially the same value of r used in [7]. See Section 1.2 for more discussion.) I.e., we prove the following theorem.

▶ **Theorem 1.** *For any $0 \leq \alpha \leq 1/2$, there is an algorithm that solves d-GDDP in time $2^{O(n^{1-2\alpha}+\log n)}$ where $d(\mathcal{L}) := n^{1/2+\alpha} \cdot \eta(\mathcal{L})$.*

Theorem 1 is primarily interesting for α strictly between zero and $1/2$. For $\alpha = 0$, Theorem 1 is outperformed by existing $2^{O(n)}$-time algorithms for CVP [16, 2]. These algorithms do not require preprocessing and are actually guaranteed to find a *closest* vector to the target $\boldsymbol{t}$, so our algorithm is beaten in many respects by the competition in this regime.

Similarly, for $\alpha = 1/2$, Babai's celebrated polynomial-time algorithm [3] always matches or outperforms Theorem 1 when instantiated with an appropriate basis as preprocessing. However, we will show in the full version [21] how to essentially combine ideas from Babai's algorithm to correct this and to beat all prior algorithms for all (constants) $0 < \alpha < 1/2$.

However, even without these improvements, this result is already quite strong for the "typical" lattices that interest us – e.g., that that arise in cryptography and those that satisfy the heuristics in [7] – which in particular satisfy $\eta(\mathcal{L}) \approx \mu(\mathcal{L})/\sqrt{n}$. In particular, Theorem 1 is already enough to remove [18]'s reliance on certain heuristic assumptions. ([18] also requires additional unrelated heuristic assumptions, which our result does not remove. We refer the reader to [18] for more information.)

Behind this result is a geometric lemma concerning the discrete Gaussian distribution that, to the author's knowledge, is novel. The lemma shows that one-dimensional projections of the discrete Gaussian look very much like a continuous Gaussian for parameters above smoothing. (See Theorem 7.)

1.2 Relation to DLW

Our presentation here is quite different from the presentation in [7]. (See also an earlier version of the same paper [10] and a closely related paper [11].) We attempt to clarify some of the differences here to avoid confusion.

First of all, DLW described their algorithm as a solution to the *Closest Vector Problem* (CVP), in which the goal is to output a vector $\boldsymbol{y} \in \mathcal{L}$ with $\|\boldsymbol{y} - \boldsymbol{t}\| \leq \gamma \cdot \text{dist}(\boldsymbol{t}, \mathcal{L})$ for some approximation factor $\gamma \geq 1$. In contrast, we call the same algorithm a GDD(P) algorithm. This discrepancy arises when one moves from heuristic algorithms to provably correct algorithms. Since $\text{dist}(\boldsymbol{t}, \mathcal{L})$ is nearly maximal for "most" $\boldsymbol{t}$ [9], DLW's heuristics quite reasonably imply that $\text{dist}(\boldsymbol{t}, \mathcal{L})$ is nearly maximal, i.e., $\text{dist}(\boldsymbol{t}, \mathcal{L}) \approx \mu(\mathcal{L})$. With this assumption, γ-CVP is essentially equivalent to $(\gamma\mu(\mathcal{L}))$-GDD. However, without such a heuristic, the two problems seem to be quite different, so that the distinction is unfortunately necessary here.

Second, since [7] describe their results in terms of CVP and do not mention the smoothing parameter $\eta(\mathcal{L})$, their results are formally incomparable with Theorem 1. However, we note that the heuristics in [7] imply that $\eta(\mathcal{L}) \approx \lambda_1(\mathcal{L})/\sqrt{n} \approx \mu(\mathcal{L})/\sqrt{n}$, and the DLW algorithm finds vectors within distance roughly $n^{\alpha}\lambda_1(\mathcal{L})$ of the target. Since we obtain vectors within distance $n^{1/2+\alpha}\eta(\mathcal{L})$, our result essentially matches theirs when their heuristics apply.

Third, while we match DLW's algorithm asymptotically, we do not claim to match the constants. Indeed, in the language of this paper, much of [7] is devoted to finding vectors within distance $c_1\sqrt{n} \cdot \eta(\mathcal{L})$ in time $2^{c_2 n + o(n)}$ for small constants $0 < c_1, c_2 < 1$. In contrast, we are mostly interested in what appears as a secondary result in that paper: the time-distance trade-off achievable for distance $n^{1/2+\alpha}\eta(\mathcal{L})$ and time $2^{O(n^{1-2\alpha} + \log n)}$ for $0 < \alpha < 1/2$. And, we make very little effort to optimize the constants. For example, [7] uses nearest neighbor data structures to let the query algorithm avoid reading the entire preprocessing, which we do not attempt to replicate here. Similarly, while [7] proposed specific techniques for computing the preprocessing in $2^{cn+o(n)}$ time, we ignore this. (We do note, however, that [1] shows how to sample the preprocessing in time $2^{n+o(n)}$.)

2 Preliminaries

Throughout this work, we adopt the common convention of expressing the running times of lattice algorithms in terms of the dimension n only, ignoring any dependence on the bit length of the input B. Formally, we should specify a particular input format for the (basis of the)

lattice (e.g., by restricting our attention to rational numbers and using the natural binary representation of a rational matrix to represent a basis for the lattice), and our running time should of course have some dependence on B. Consideration of the bit length would simply add a poly(B) factor to the running time for the algorithm(s) considered in this paper, provided that the input format allows for efficient arithmetic operations.

2.1 The discrete Gaussian

For a vector $\boldsymbol{x} \in \mathbb{R}^n$ and parameter $s > 0$, we write $\rho_s(\boldsymbol{x}) := \exp(-\pi\|\boldsymbol{x}\|^2/s^2)$ for the Gaussian mass of $\boldsymbol{x}$ with parameter s. For a lattice $\mathcal{L} \subset \mathbb{R}^n$ and shift vector $\boldsymbol{t} \in \mathbb{R}^n$, we write

$$\rho_s(\mathcal{L} - \boldsymbol{t}) := \sum_{\boldsymbol{y} \in \mathcal{L}} \rho_s(\boldsymbol{y} - \boldsymbol{t})$$

for the Gaussian mass of $\mathcal{L} - \boldsymbol{t}$ with parameter s. We write $D_{\mathcal{L},s}$ for the probability distribution over $\mathcal{L}$ defined by

$$\Pr_{\boldsymbol{X} \sim D_{\mathcal{L},s}}[\boldsymbol{X} = \boldsymbol{y}] = \frac{\rho_s(\boldsymbol{y})}{\rho_s(\mathcal{L})}$$

for $\boldsymbol{y} \in \mathcal{L}$.

The dual lattice $\mathcal{L}^* \subset \mathbb{R}^n$ is the set of vectors that have integer inner product with all lattice vectors,

$$\mathcal{L}^* := \{\boldsymbol{w} \in \mathbb{R}^n \ : \ \forall \boldsymbol{y} \in \mathcal{L},\ \langle \boldsymbol{w}, \boldsymbol{y} \rangle \in \mathbb{Z}\} .$$

Micciancio and Regev defined the *smoothing parameter* $\eta(\mathcal{L})$ as the unique parameter s such that $\rho_{1/s}(\mathcal{L}^*) = 3/2$ [15].[1] The following claim justifies the name "smoothing parameter," and it is the only fact about the smoothing parameter that we will need.

▷ Claim 2. For any lattice $\mathcal{L} \subset \mathbb{R}^n$, parameter $s \geq \eta(\mathcal{L})$, and shift $\boldsymbol{t} \in \mathbb{R}^n$,

$$\frac{1}{3} \leq \frac{\rho_s(\mathcal{L} - \boldsymbol{t})}{\rho_s(\mathcal{L})} \leq 1 .$$

We will also need a simplified version of Banaszczyk's celebrated tail bound for the discrete Gaussian [4].

▶ **Theorem 3.** *For any lattice $\mathcal{L} \subset \mathbb{R}^n$ and parameter $s > 0$,*

$$\Pr_{\boldsymbol{X} \sim D_{\mathcal{L},s}}[\|\boldsymbol{X}\| \geq \sqrt{n}s] \leq 2^{-n} .$$

Finally, we will need the following rather weak consequence of Babai's algorithm [3].

▶ **Theorem 4.** *There is a polynomial-time algorithm for $(2^n\eta(\mathcal{L}))$-GDD.*

2.2 ε-nets

For $\varepsilon > 0$, we say that a set $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_M\} \subset \mathbb{R}^n$ of unit vectors with $\|\boldsymbol{v}_i\| = 1$ is an ε-net of the unit sphere if for any $\boldsymbol{t} \in \mathbb{R}^n$ with $\|\boldsymbol{t}\| = 1$, there exists $\boldsymbol{v}_i$ such that $\|\boldsymbol{v}_i - \boldsymbol{t}\| \leq \varepsilon$. We will use a simple bound on the size of such a net, which can be proven via a simple packing argument. See [22, Lemma 5.2], for example.

▶ **Lemma 5.** *For any $\varepsilon > 0$, there exists an ε-net of the unit sphere in $\mathbb{R}^n$ with $(1 + 2/\varepsilon)^n$ points.*

[1] This is more commonly referred to as $\eta_{1/2}(\mathcal{L})$, where $\eta_\varepsilon(\mathcal{L})$ is the unique parameter s such that $\rho_{1/s}(\mathcal{L}^*) = 1 + \varepsilon$. Since we will always take $\varepsilon = 1/2$, we simply omit it. Our results remain essentially unchanged if we take ε to be any constant strictly between zero and one.

3 The algorithm

We consider the following algorithm for GDDP. For an input lattice $\mathcal{L} \subset \mathbb{R}^n$ with $n \geq 40$, the preprocessing consists of N lattice vectors $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_N \in \mathcal{L}$. On input $\boldsymbol{t} \in \mathbb{R}^n$, the query algorithm behaves as follows. It first uses Theorem 4 to find $\boldsymbol{t}_0 \in \mathcal{L} + \boldsymbol{t}$ such that $\|\boldsymbol{t}_0\| \leq 2^n \eta(\mathcal{L})$ and sets $j = 0$. The algorithm then does the following repeatedly. It finds an index i and integer k such that $\|\boldsymbol{t}_j - k\boldsymbol{y}_i\|^2 \leq (1 - 1/n^2) \cdot \|\boldsymbol{t}_j\|^2$, sets $\boldsymbol{t}_{j+1} := \boldsymbol{t}_j - k\boldsymbol{y}_i$, and increments j. Once the algorithm fails to find such a vector, it outputs $\boldsymbol{t}_j - \boldsymbol{t} \in \mathcal{L}$.[2]

Our main theorem shows that this algorithm will succeed when the preprocessing is chosen from the right distribution. We emphasize the order of quantifiers: with high probability over the preprocessing, the algorithm works *for all targets* $\boldsymbol{t} \in \mathbb{R}^n$. In particular, there exists fixed preprocessing that works for all targets $\boldsymbol{t}$.

▶ **Theorem 6.** *For any α with $\frac{2}{\log n} \leq \alpha \leq \frac{1}{2}$, when the preprocessing of the above algorithm consists of $N_\alpha := n^2 e^{(n^{1/2-\alpha}+4)^2} = 2^{O(n^{1-2\alpha}+\log n)}$ samples from $D_{\mathcal{L},s}$ for $s := \eta(\mathcal{L})$, it yields a solution to d-GDDP in time $\mathrm{poly}(n) \cdot N_\alpha$ with high probability, where $d := n^{1/2+\alpha} \cdot \eta(\mathcal{L})$.*

Proof. By scaling appropriately, we may assume without loss of generality that $d = 1$, and therefore $s = n^{-1/2-\alpha}$. Let $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_{N_\alpha} \sim D_{\mathcal{L},s}$. To prove correctness, we must show that, with high probability over the $\boldsymbol{y}_i$, for every $\boldsymbol{t} \in \mathbb{R}^n$ with $\|\boldsymbol{t}\| \geq 1$, there exists an index i and integer k such that $\|\boldsymbol{t} - k\boldsymbol{y}_i\|^2 \leq (1 - 1/n^2) \cdot \|\boldsymbol{t}\|^2$. It suffices to prove that for $\|\boldsymbol{t}\| = 1$, there exists an i with $\|\boldsymbol{t} - \boldsymbol{y}_i\|^2 \leq 1 - 4/n^2$.[3] Finally, to prove *this*, it suffices to take a $(1/n^3)$-net of the unit sphere, $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_M$, and to show that for each j, there exists an i such that $\|\boldsymbol{v}_j - \boldsymbol{y}_i\|^2 \leq 1 - 5/n^2$.

By Lemma 5, there exists such a net of cardinality $M = (3n)^{3n}$. For each $\boldsymbol{v}_j$ in this net and each index i, we have by Corollary 8 (proven below) that

$$\begin{aligned}\Pr\left[\|\boldsymbol{v}_j - \boldsymbol{y}_i\|^2 \leq 1 - 5/n^2\right] &\geq \exp(-\pi(5/(n^2 s) + ns + 4)^2/4) - 2^{-n} \\ &\geq \exp(-(n^{1/2-\alpha} + 4)^2) \\ &= n^2/N_\alpha \, .\end{aligned}$$

Since the $\boldsymbol{y}_i$ are sampled independently, the probability that no such i exists is at most $(1 - n^2/N_\alpha)^{N_\alpha} < 2^{-n}/M$. The result then follows by taking a union bound over the $\boldsymbol{v}_j$. ◀

3.1 One-dimensional projections of the discrete Gaussian

We are interested in the lower bound in the following lemma (whose proof uses a very nice idea from [13]). The upper bound (i.e., the subgaussianity of the discrete Gaussian) applies for all parameters $s > 0$ and is well known. (It appears in slightly different forms in [5, 17]. See also [14, Lemma 2.8].) We only include the upper bound for comparison, and we make no effort to optimize the lower-order term.

---

[2] The author made no effort to optimize these parameters. Notice that we can find such an i and k (if they exist) in time essentially $\mathrm{poly}(n) \cdot N$. To guarantee a total running time of $\mathrm{poly}(n) \cdot N$, we can also assume that the algorithm halts and outputs $\boldsymbol{t}_j - \boldsymbol{t} \in \mathcal{L}$ if j reaches, say, $100n^3$. This is not strictly necessary, since we will have $\|\boldsymbol{y}_i\| \approx \sqrt{n} \cdot \eta(\mathcal{L})$ with very high probability.

[3] Indeed, suppose that $\|\boldsymbol{t} - \boldsymbol{y}_i\|^2 \leq 1 - 4/n^2$ and $\|\boldsymbol{t}\| = 1$, so that in particular $\langle \boldsymbol{y}_i, \boldsymbol{t} \rangle \geq 0$. Then for any $\beta/2 \leq k \leq \beta$,

$$\frac{\|\beta\boldsymbol{t} - k\boldsymbol{y}_i\|^2}{\|\beta\boldsymbol{t}\|^2} = 1 - \frac{k^2}{\beta^2} \cdot \left(1 - \|\boldsymbol{t} - \boldsymbol{y}_i\|^2\right) - \left(\frac{2k}{\beta} - \frac{2k^2}{\beta^2}\right) \cdot \langle \boldsymbol{y}_i, \boldsymbol{t} \rangle \leq 1 - \frac{k^2}{\beta^2} \cdot \frac{4}{n^2} \leq 1 - \frac{1}{n^2} \, .$$

▶ **Theorem 7.** *For any lattice $\mathcal{L} \subset \mathbb{R}^n$, parameter $s \geq \eta(\mathcal{L})$, unit vector $\boldsymbol{v} \in \mathbb{R}^n$ with $\|\boldsymbol{v}\| = 1$, and $r_0 > 0$, we have*

$$\exp(-\pi(r_0/s+2)^2) < \Pr_{\boldsymbol{X}\sim D_{\mathcal{L},s}}\left[\langle \boldsymbol{X}, \boldsymbol{v}\rangle \geq r_0\right] \leq \exp(-\pi r_0^2/s^2)\ . \tag{1}$$

Before presenting the proof, we provide some of the intuition behind it. The idea is to control the moment-generating function $g(\beta) := \mathbb{E}[\exp(2\pi\beta\langle \boldsymbol{X}, \boldsymbol{v}\rangle)]$ for $\beta > 0$. For a continuous Gaussian, this is $\exp(\pi\beta^2)$, and the discrete Gaussian behaves similarly, with $g(\beta) \approx \exp(\pi\beta^2)$ (see Eq. (2)). For a fixed value of β, knowledge of $g(\beta)$ is insufficient to prove something like Eq. (1). So, we take a weighted combination $\alpha_1 g(\beta_1) - \alpha_2 g(\beta_2) - \alpha_3 g(\beta_3)$ for appropriately chosen weights to essentially approximate

$$\mathbb{E}[1_{\langle \boldsymbol{X}, \boldsymbol{v}\rangle \gtrsim r_0}] \approx \Pr_{\boldsymbol{X}\sim D_{\mathcal{L},s}}\left[\langle \boldsymbol{X}, \boldsymbol{v}\rangle \geq r_0\right]\ .$$

More specifically, we define the function $f(r)$ as in Eq. (3), which satisfies $f(r) < 0$ unless $r \approx r_0$ and $f(r) \approx \exp(2\pi\beta r_0)$ for $r \approx r_0$. We then use our bounds on the moment-generating function to show that $\mathbb{E}[f(\langle \boldsymbol{X}, \boldsymbol{v}\rangle)]$ is not too small, which implies the result.

Proof. By scaling appropriately, we may assume that $s = 1$. Let $\beta > 0$ to be chosen later. By completing the square in the exponent, we see that

$$\mathbb{E}[\exp(2\pi\beta\langle \boldsymbol{X}, \boldsymbol{v}\rangle)] = \exp(\pi\beta^2) \cdot \frac{\rho_1(\mathcal{L} - \beta\boldsymbol{v})}{\rho_1(\mathcal{L})}\ .$$

Therefore, by Claim 2,

$$\frac{1}{3} \leq \exp(-\pi\beta^2) \cdot \mathbb{E}[\exp(2\pi\beta\langle \boldsymbol{X}, \boldsymbol{v}\rangle)] \leq 1\ . \tag{2}$$

I.e., we know the moment generating function of $\langle \boldsymbol{X}, \boldsymbol{v}\rangle$ to within a multiplicative constant. The upper bound in Eq. (1) then follows from taking $\beta = r_0$ and applying Markov's inequality. (This proof of the upper bound is identical to the proof in [14]. See their Lemma 2.8 and their discussion above it.)

Turning to the lower bound, for $r \in \mathbb{R}$, let

$$f(r) := \exp(2\pi\beta r) \cdot \big(1 - \exp(2\pi(r_0 - r)) - \exp(2\pi(r - r_0 - 2))\big)\ . \tag{3}$$

Notice that $f(r) < 0$ unless $r_0 < r < r_0 + 2$. And, $f(r) < \exp(2\pi\beta r)$, which together with the previous two inequalities implies that $f(r) < \exp(2\pi\beta(r_0+2))$ for all r. Therefore,

$$\mathbb{E}\left[f(\langle \boldsymbol{X}, \boldsymbol{v}\rangle)\right] < \exp(2\pi\beta(r_0+2)) \cdot \Pr\left[\langle \boldsymbol{X}, \boldsymbol{v}\rangle \geq r_0\right]\ . \tag{4}$$

By applying Eq. (2) term-wise and taking $\beta = r_0 + 1$, we have

$$\begin{aligned}
\mathbb{E}\left[f(\langle \boldsymbol{X}, \boldsymbol{v}\rangle)\right] &= \mathbb{E}\left[\exp(2\pi\beta\langle \boldsymbol{X}, \boldsymbol{v}\rangle)\right] \\
&\quad - \exp(2\pi r_0)\,\mathbb{E}\left[\exp(2\pi(\beta-1)\langle \boldsymbol{X}, \boldsymbol{v}\rangle)\right] \\
&\quad - \exp(-2\pi(r_0+2))\,\mathbb{E}\left[\exp(2\pi(\beta+1)\langle \boldsymbol{X}, \boldsymbol{v}\rangle)\right] \\
&\geq \frac{1}{3}\cdot\exp(\pi\beta^2) - \exp(\pi(\beta-1)^2 + 2\pi r_0) - \exp(\pi(\beta+1)^2 - 2\pi(r_0+2)) \\
&= \exp(\pi r_0^2 + 2\pi r_0)\cdot(e^\pi/3 - 2) \\
&> \exp(\pi r_0^2 + 2\pi r_0)\ .
\end{aligned} \tag{5}$$

Combining Eqs. (4) and (5) and rearranging, we have

$$\Pr\left[\langle \boldsymbol{X}, \boldsymbol{v}\rangle \geq r_0\right] > \exp(\pi r_0^2 + 2\pi r_0 - 2\pi\beta(r_0+2)) = \exp(-\pi(r_0+2)^2)\ ,$$

as needed. ◀

▶ **Corollary 8.** *For any $0 < r < 1$, lattice $\mathcal{L} \subset \mathbb{R}^n$, unit vector $\boldsymbol{v} \in \mathbb{R}^n$ with $\|\boldsymbol{v}\| = 1$, and $s \geq \eta(\mathcal{L})$, we have*

$$\Pr_{\boldsymbol{X} \sim D_{\mathcal{L},s}} \left[\|\boldsymbol{v} - \boldsymbol{X}\|^2 \leq 1 - r\right] > \exp(-\pi(r/s + ns + 4)^2/4) - 2^{-n} .$$

Proof. Notice that $\|\boldsymbol{v}-\boldsymbol{X}\|^2 = 1+\|\boldsymbol{X}\|^2-2\langle \boldsymbol{v}, \boldsymbol{X}\rangle$. By Theorem 3, we have that $\|\boldsymbol{X}\|^2 \leq ns^2$ except with probability at most 2^{-n}. By Theorem 7, we see that

$$\Pr[\langle \boldsymbol{v}, \boldsymbol{X}\rangle \geq (ns^2 + r)/2] > \exp(-\pi(r/s + ns + 4)^2/4) .$$

The result follows from union bound. ◀

References

1. Divesh Aggarwal, Daniel Dadush, Oded Regev, and Noah Stephens-Davidowitz. Solving the Shortest Vector Problem in 2^n time via Discrete Gaussian Sampling. In *STOC*, 2015. `arXiv:abs/1412.7994`.
2. Divesh Aggarwal, Daniel Dadush, and Noah Stephens-Davidowitz. Solving the Closest Vector Problem in 2^n time–The discrete Gaussian strikes again! In *FOCS*, 2015. `arXiv:1504.01995`.
3. L. Babai. On Lovász' lattice reduction and the nearest lattice point problem. *Combinatorica*, 6(1), 1986.
4. Wojciech Banaszczyk. New bounds in some transference theorems in the geometry of numbers. *Mathematische Annalen*, 296(4), 1993.
5. Wojciech Banaszczyk. Inequalites for convex bodies and polar reciprocal lattices in $\mathbb{R}^n$. *Discrete & Computational Geometry*, 13, 1995.
6. Daniel Dadush, Oded Regev, and Noah Stephens-Davidowitz. On the Closest Vector Problem with a distance guarantee. In *CCC*, 2014. `arXiv:1409.8063`.
7. Emmanouil Doulgerakis, Thijs Laarhoven, and Benne de de Weger. Finding closest lattice vectors using approximate Voronoi cells. In *PQCrypto*, 2019. (To appear.) `https://eprint.iacr.org/2016/888`.
8. Nicolas Gama and Phong Q. Nguyen. Finding Short Lattice Vectors Within Mordell's Inequality. In *STOC*, 2008.
9. Ishay Haviv, Vadim Lyubashevsky, and Oded Regev. A Note on the Distribution of the Distance from a Lattice. *Discrete & Computational Geometry*, 41(1), 2009.
10. Thijs Laarhoven. Finding closest lattice vectors using approximate Voronoi cells, 2016. URL: `https://eprint.iacr.org/2016/888/20161219:141310`.
11. Thijs Laarhoven. Sieving for Closest Lattice Vectors (with Preprocessing). In *SAC*, 2016.
12. Yi-Kai Liu, Vadim Lyubashevsky, and Daniele Micciancio. On Bounded Distance Decoding for general lattices. In *RANDOM*, 2006.
13. J. E. Mazo and A. M. Odlyzko. Lattice points in high-dimensional spheres. *Monatshefte für Mathematik*, 110(1), 1990.
14. Daniele Micciancio and Chris Peikert. Trapdoors for Lattices: Simpler, Tighter, Faster, Smaller. In *EUROCRYPT*, 2012. URL: `https://eprint.iacr.org/2011/501`.
15. Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on Gaussian measures. *SIAM Journal of Computing*, 37(1), 2007.
16. Daniele Micciancio and Panagiotis Voulgaris. A deterministic single exponential time algorithm for most lattice problems based on Voronoi cell computations. *SIAM Journal on Computing*, 42(3), 2013.
17. Chris Peikert. Limits on the Hardness of Lattice Problems in ℓ_p Norms. *Computational Complexity*, 17(2), 2008.
18. Alice Pellet–Mary, Guillaume Hanrot, and Damien Stehlé. Approx-SVP in Ideal Lattices with Pre-processing. In *Eurocrypt*, 2019. (to appear).

19 Claus-Peter Schnorr. A Hierarchy of Polynomial Time Lattice Basis Reduction Algorithms. *Theor. Comput. Sci.*, 53(23), 1987.

20 Noah Stephens-Davidowitz. *On the Gaussian measure over lattices.* Ph.D. thesis, New York University, 2017.

21 Noah Stephens-Davidowitz. A time-distance trade-off for GDD with preprocessing – Instantiating the DLW heuristic, 2019. `arXiv:1902.08340`.

22 Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. In *Compressed Sensing: Theory and Applications*. Cambridge University Press, 2012.

Limits on the Universal Method for Matrix Multiplication

Josh Alman
MIT CSAIL and EECS, Cambridge, MA, USA
jalman@mit.edu

Abstract

In this work, we prove limitations on the known methods for designing matrix multiplication algorithms. Alman and Vassilevska Williams [2] recently defined the *Universal Method*, which substantially generalizes all the known approaches including Strassen's Laser Method [20] and Cohn and Umans' Group Theoretic Method [9]. We prove concrete lower bounds on the algorithms one can design by applying the Universal Method to many different tensors. Our proofs use new tools for upper bounding the *asymptotic slice rank* of a wide range of tensors. Our main result is that the Universal method applied to any Coppersmith-Winograd tensor CW_q cannot yield a bound on ω, the exponent of matrix multiplication, better than 2.16805. By comparison, it was previously only known that the weaker "Galactic Method" applied to CW_q could not achieve an exponent of 2.

We also study the Laser Method (which is, in principle, a highly special case of the Universal Method) and prove that it is "complete" for matrix multiplication algorithms: when it applies to a tensor T, it achieves $\omega = 2$ if and only if it is possible for the Universal method applied to T to achieve $\omega = 2$. Hence, the Laser Method, which was originally used as an algorithmic tool, can also be seen as a lower bounding tool. For example, in their landmark paper, Coppersmith and Winograd [12] achieved a bound of $\omega \leq 2.376$, by applying the Laser Method to CW_q. By our result, the fact that they did not achieve $\omega = 2$ *implies* a lower bound on the Universal Method applied to CW_q. Indeed, if it were possible for the Universal Method applied to CW_q to achieve $\omega = 2$, then Coppersmith and Winograd's application of the Laser Method would have achieved $\omega = 2$.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms; Theory of computation → Algebraic complexity theory

Keywords and phrases Matrix Multiplication, Laser Method, Slice Rank

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.12

Funding *Josh Alman*: Supported by two NSF Career Awards.

Acknowledgements I would like to thank Matthias Christandl, Joshua Grochow, Ryan Williams, Virginia Vassilevska Williams, and Jeroen Zuiddam for helpful discussions and suggestions.

1 Introduction

One of the biggest open questions in computer science asks how quickly one can multiply two matrices. Progress on this problems is measured by giving bounds on ω, the *exponent of matrix multiplication*, defined as the smallest real number such that two $n \times n$ matrices over a field can be multiplied using $n^{\omega+o(1)}$ field operations. Since Strassen's breakthrough algorithm [21] showing that $\omega \leq \log_2(7) \approx 2.81$, there has been a long line of work, resulting in the current best bound of $\omega \leq 2.3729$ [26, 18], and it is popularly conjectured that $\omega = 2$.

The key to Strassen's algorithm is an algebraic identity showing how $2 \times 2 \times 2$ matrix multiplication can be computed surprisingly efficiently (in particular, Strassen showed that the $2 \times 2 \times 2$ matrix multiplication tensor has rank at most 7; see Section 3 for precise definitions). Arguing about the ranks of larger matrix multiplication tensors has proven to be quite difficult – in fact, even the rank of the $3 \times 3 \times 3$ matrix multiplication tensor

isn't currently known. Progress on bounding ω since Strassen's algorithm has thus taken the following approach: Pick a tensor (trilinear form) T, typically not a matrix multiplication tensor, such that

- Powers $T^{\otimes n}$ of T can be efficiently computed (i.e. T has low asymptotic rank), and
- T is useful for performing matrix multiplication, since large matrix multiplication tensors can be "embedded" within powers of T.

Combined, these give an upper bound on the rank of matrix multiplication itself, and hence ω.

The most general type of embedding which is known to preserve the ranks of tensors as required for the above approach is a *degeneration*. In [2], the author and Vassilevska Williams called this method of taking a tensor T and finding the best possible degeneration of powers $T^{\otimes n}$ into matrix multiplication tensors the *Universal Method applied to* T, and the best bound on ω which can be proved in this way is written $\omega_u(T)$. They also defined two weaker methods: *the Galactic Method applied to* T, in which the "embedding" must be a more restrictive *monomial degeneration*, resulting in the bound $\omega_g(T)$ on ω, and *the Solar Method applied to* T, in which the "embedding" must be an even more restrictive *zeroing out*, resulting in the bound $\omega_s(T)$ on ω. Since monomial degenerations and zeroing outs are successively more restrictive types of degenerations, we have that for all tensors T,

$$\omega \leq \omega_u(T) \leq \omega_g(T) \leq \omega_s(T).$$

These methods are *very general*; there are no known methods for computing $\omega_u(T)$, $\omega_g(T)$, or $\omega_s(T)$ for a given tensor T, and these quantities are even unknown for very well-studied tensors T. The two main approaches to designing matrix multiplication algorithms are the Laser Method of Strassen [20] and the Group-Theoretic Method of Cohn and Umans [9]. Both of these approaches show how to give upper bounds on $\omega_s(T)$ for particular structured tensors T (and hence upper bound ω itself). In other words, they both give ways to find zeroing outs of tensors into matrix multiplication tensors, but not necessarily the best zeroing outs. In fact, it is known that the Laser Method does not always give the best zeroing out for a particular tensor T, since the improvements from [12] to later works [13, 26, 18] can be seen as giving slight improvements to the Laser Method to find better and better zeroing outs[1]. The Group-Theoretic Method, like the Solar Method, is very general, and it is not clear how to optimally apply it to a particular group or family of groups.

All of the improvements on bounding ω for the past 30+ years have come from studying the Coppersmith-Winograd family of tensors $\{CW_q\}_{q\in\mathbb{N}}$. The Laser Method applied to powers of CW_5 gives the bound $\omega_s(CW_5) \leq 2.3729$. The Group-Theoretic Method can also prove the best known bound $\omega \leq 2.3729$, by simulating the Laser Method analysis of CW_q (see e.g. [1] for more details). Despite a long line of work on matrix multiplication, there are no known tensors[2] which seem to come close to achieving the bounds one can obtain using CW_q. This leads to the first main question of this paper:

▶ **Question 1.** How much can we improve our bound on ω using a more clever analysis of the Coppersmith-Winograd tensor?

The author and Vassilevska Williams [2] addressed this question by showing that there is a constant $c > 2$ so that for all q, $\omega_g(CW_q) > c$. In other words, the Galactic Method (monomial degenerations) cannot be used with CW_q to prove $\omega = 2$. However, this leaves open

[1] These works apply the Laser Method to higher powers of the tensor $T = CW_q$, a technique which is still captured by the Solar Method.

[2] The author and Vassilevska Williams [2] study a generalization of CW_q which can tie the best known bound, but its analysis is identical to that of CW_q. Our lower bounds in this paper will apply equally well to this generalized class as to CW_q itself.

a number of important questions: How close to 2 can we get using monomial degenerations; could it be that $\omega_g(CW_q) \leq 2.1$? Perhaps more importantly, what if we are allowed to use arbitrary degenerations; could it be that $\omega_u(CW_q) \leq 2.1$, or even $\omega_u(CW_q) = 2$?

The second main question of this paper concerns the Laser Method. The Laser Method upper bounds $\omega_s(T)$ for any tensor T with certain structure (which we describe in detail in Section 6), and has led to every improvement on ω since its introduction by Strassen [20].

▶ **Question 2.** When the Laser Method applies to a tensor T, how close does it come to optimally analyzing T?

As discussed, we know the Laser Method does not always give a tight bound on $\omega_s(T)$. For instance, Coppersmith-Winograd [12] applied the Laser Method to CW_q to prove $\omega_s(CW_q) \leq 2.376$, and then later work [13, 26, 18] analyzed higher and higher powers of CW_q to show $\omega_s(CW_q) \leq 2.373$. Ambainis, Filmus and Le Gall [3] showed that analyzing higher and higher powers of CW_q itself with the Laser Method cannot yield an upper bound better than $\omega_s(CW_q) \leq 2.3725$. What about for other tensors? Could there be a tensor such that applying the Laser Method to T yields $\omega_s(T) \leq c$ for some $c > 2$, but applying the Laser Method to high powers $T^{\otimes n}$ of T yields $\omega_s(T) = 2$? Could applying an entirely different method to such a T, using arbitrary degenerations and not just zeroing outs, show that $\omega_u(T) = 2$?

1.1 Our Results

We give strong resolutions to both Question 1 and Question 2.

Universal Method Lower Bounds

To resolve Question 1, we prove a new lower bound for the Coppersmith-Winograd tensor:

▶ **Theorem 3.** $\omega_u(CW_q) \geq 2.16805$ *for all* q.

In other words, no analysis of CW_q, using any techniques within the Universal Method, can prove a bound on ω better than 2.16805. This generalizes the main result of [2] from the Galactic method to the *Universal* method, and gives a more concrete lower bound, increasing the bound from "a constant greater than 2" to 2.16805. We also give stronger lower bounds for particular tensors in the family. For instance, for the specific tensor CW_5 which yields the current best bound on ω, we show $\omega_u(CW_5) \geq 2.21912\ldots$.

Our proof of Theorem 3 proceeds by upper bounding $\tilde{S}(CW_q)$, the *asymptotic slice rank* of CW_q. The slice rank of a tensor, denoted $S(T)$, was first introduced by Blasiak et al. [5] in the context of lower bounds against the Group-Theoretic Method. In order to study degenerations of *powers* of tensors, rather than just tensors themselves, we need to study an *asymptotic* version of slice rank, $\tilde{S}$. This is important since the slice rank of a product of two tensors can be greater than the product of their slice ranks, and as we will show, $S(CW_q^{\otimes n})$ is much greater than $S(CW_q)^n$ for big enough n.

We will give three different tools for proving upper bounds on $\tilde{S}(T)$ for many different tensors T. These, combined with the known connection, that upper bounds on the slice rank of T yield lower bounds on $\omega_u(T)$, will imply our lower bound for CW_q as well as many other tensors of interest, including: the same lower bound $\omega_u(CW_{q,\sigma}) \geq 2.16805$ for any *generalized Coppersmith-Winograd tensor* $CW_{q,\sigma}$ as introduced in [2], a similar lower bound for $cw_{q,\sigma}$, the generalized "simple" Coppersmith-Winograd tensor missing its "corner terms", and a lower bound for T_q, the structural tensor of the cyclic group C_q, matching the lower bounds obtained by [1, 5]. In Section 5 we give tables of our precise lower bounds for these and other tensors.

The Galactic Method lower bounds of [2] were proved by introducing a suite of tools for giving upper bounds on $\tilde{I}(T)$, the *asymptotic independence number* (sometimes also called the "galactic subrank" or the "monomial degeneration subrank") for many tensors T. We will show that our new tools are able to prove at least as high a lower bound on $\tilde{S}(T)$ as the tools of [2] can prove on $\tilde{I}(T)$. We thus show that all of those previously known Galactic Method lower bounds hold for the Universal Method as well.

We also show how our slice rank lower bounds can be used to study other properties of tensors. Coppersmith and Winograd [12] introduced the notion of the *value* $V_\tau(T)$ of a tensor T, which is useful when applying the Laser Method to a larger tensor T' which contains T as a subtensor. We show how our slice rank lower bounding tools yield a tight upper bound on the value of t_{112}, the notorious subtensor of $CW_q^{\otimes 2}$ which arises when applying the Laser Method to powers of CW_q. Although the value $V_\tau(t_{112})$ appears in every analysis of CW_q since [12], including [13, 26, 18, 17, 14], the best lower bound on it has not improved since [12], and our new upper bound here helps explain why. See Sections 3.5 and 5.4 for more details.

We briefly note that our lower bound of $2.16805 > 2 + \frac{1}{6}$ in Theorem 3 may be significant when compared to the recent algorithm of Cohen, Lee and Song [8] which solves n-variable linear programs in time about $O(n^\omega + n^{2+1/6})$.

The Laser Method is "Complete"

We also show that for a wide class of tensors T, including CW_q, cw_q, T_q, and all the other tensors we study in Section 5, our tools are tight, meaning they not only give an upper bound on $\tilde{S}(T)$, but they also give a matching lower bound. Hence, for these tensors T, no better lower bound on $\omega_u(T)$ is possible by arguing only about $\tilde{S}(T)$.

The tensors we prove this for are what we call *laser-ready* tensors – tensors to which the Laser Method (as used by [12] on CW_q) applies; see Definition 25 for the precise definition. Tensors need certain structure to be laser-ready, but tensors T with this structure are essentially the only ones for which successful techniques for upper bounding $\omega_u(T)$ are known. In fact, every record-holding tensor in the history of matrix multiplication algorithm design has been laser-ready.

We show that for any laser-ready tensor T, the Laser Method can be used to construct a degeneration from $T^{\otimes n}$ to an independent tensor of size $\Lambda^{n-o(n)}$, where Λ is the upper bound on $\tilde{S}(T)$ implied by one of our tools, Theorem 18. Combined, these imply that $\tilde{S}(T) = \Lambda$, showing that the lower bound from Theorem 18 is tight. This gives an intriguing answer to Question 2:

▶ **Theorem 4.** *If T is a laser-ready tensor, and the Laser Method applied to T yields the bound $\omega_u(T) \leq c$ for some $c > 2$, then $\omega_u(T) > 2$.*

To reiterate: If T is any tensor to which the Laser Method applies (as in Definition 25), and the Laser Method does not yield $\omega = 2$ when applied to T, then in fact $\omega_u(T) > 2$, and even the substantially more general Universal method applied to T cannot yield $\omega = 2$. Hence, the Laser Method, which was originally used as an algorithmic tool, can also be seen as a lower bounding tool. Conversely, Theorem 4 shows that the Laser Method is "complete", in the sense that it cannot yield a bound on ω worse than 2 when applied to a tensor which is able to prove $\omega = 2$.

Theorem 4 explains and generalizes a number of phenomena:

- The fact that Coppersmith-Winograd [12] applied the Laser method to the tensor CW_q and achieved an upper bound greater than 2 on ω *implies that* $\omega_u(CW_q) > 2$, and no *arbitrary degeneration* of powers of CW_q can yield $\omega = 2$.

- As mentioned above, it is known that applying the Laser method to higher and higher powers of a tensor T can successively improve the resulting upper bound on ω. Theorem 4 shows that if the Laser method applied to the first power of any tensor T did not yield $\omega = 2$, then this sequence of Laser method applications (which is a special case of the Universal method) must converge to a value greater than 2 as well. This generalizes the result of Ambainis, Filmus and Le Gall [3], who proved this about applying the Laser Method to higher and higher powers of the specific tensor $T = CW_q$.
- Our result also generalizes the result of Kleinberg, Speyer and Sawin [16], where it was shown that (what can be seen as) the Laser method achieves a tight lower bound on $\tilde{S}(T_q^{lower})$, matching the upper bound of Blasiak et al. [5]. Indeed, T_q^{lower}, the lower triangular part of T_q, is a laser-ready tensor.

Our proof of Theorem 4 also sheds light on a notion related to the asymptotic slice rank $\tilde{S}(T)$ of a tensor T, called the *asymptotic subrank* $\tilde{Q}(T)$ of T. $\tilde{Q}$ is a "dual" notion of asymptotic rank, and it is important in the definition of Strassen's asymptotic spectrum of tensors [20].

It is not hard to see (and follows, for instance, from Propositions 8 and 9 below) that $\tilde{Q}(T) \leq \tilde{S}(T)$ for all tensors T. However, there are no known separations between the two notions; whether there exists a tensor T such that $\tilde{Q}(T) < \tilde{S}(T)$ is an open question. As a Corollary of Theorem 4, we prove:

▶ **Corollary 5.** *Every laser-ready tensor T has $\tilde{Q}(T) = \tilde{S}(T)$.*

Since, as discussed above, almost all of the most-studied tensors are laser-ready, this might help explain why we have been unable to separate the two notions.

1.2 Other Related Work

Probabilistic Tensors and Support Rank

Cohn and Umans [10] introduced the notion of the *support rank* of tensors, and showed that upper bounds on the support rank of matrix multiplication tensors can be used to design faster *Boolean* matrix multiplication algorithms. Recently, Karppa and Kaski [15] used "probabilistic tensors" as another way to design Boolean matrix multiplication algorithms.

In fact, our tools for proving asymptotic slice rank upper bounds can be used to prove lower bounds on these approaches as well. For instance, our results imply that finding a "weighted" matrix multiplication tensor as a degeneration of a power of CW_q (in order to prove a support rank upper bound) cannot result in a better exponent for Boolean matrix multiplication than 2.16805.

This is because "weighted" matrix multiplication tensors can degenerate into independent tensors just as large as their unweighted counterparts. Similarly, if a probabilistic tensor $\mathcal{T}$ is degenerated into a (probabilistic) matrix multiplication tensor, Karppa and Kaski show that this gives a corresponding support rank expression for matrix multiplication as well, and so upper bounds on $\tilde{S}(T)$ for any T in the support of $\mathcal{T}$ also result in lower bounds on this approach.

Concurrent Work

Christandl, Vrana and Zuiddam [6] independently proved some of the same lower bounds on ω_u as us, including Theorem 3. Although we achieve the same upper bounds on $\omega_u(T)$ for a number of tensors, our techniques seem different: we use simple combinatorial tools

generalizing those from our prior work [2], while their bounds use the seemingly more complicated machinery of Strassen's asymptotic spectrum of tensors [23]. They thus phrase their results in terms of the asymptotic subrank $\tilde{Q}(T)$ of tensors rather than the asymptotic slice rank $\tilde{S}(T)$, and the fact that their bounds are often the same as ours is related to the fact we prove, in Corollary 5, that $\tilde{Q}(T) = \tilde{S}(T)$ for all of the tensors we study; see the bottom of Section 3.6 for a more technical discussion of the differences between the two notions. Our other results and applications of our techniques are, as far as we know, entirely new, including our matching lower bounds for $\tilde{S}(CW_q)$, $\tilde{S}(cw_q)$, and $\tilde{S}(T_q)$, bounding the value $V_\tau(T)$ of tensors, and all our results about the completeness of the Laser Method. By comparison, their "irreversibility" approach only seems to upper bound $\omega_u(T)$ itself.

1.3 Outline

In Section 2 we give an overview of the proofs of our main results. In Section 3 we introduce all the concepts and notation related to tensors which will be used throughout the paper. In particular, in Subsection 3.6 we introduce the relevant notions and basic properties related to slice rank. In Section 4 we present the proofs of our new lower bounding tools for asymptotic slice rank. In Section 5 we apply these tools to a number of tensors of interest including CW_q. Finally, in Section 6, we define and discuss the "completeness" of the Laser method.

2 Proof Overview

We give a brief overview of the techniques we use to prove our main results, Theorems 3 and 4. All the technical terms we refer to here will be precisely defined in Section 3.

Section 3.6: Asymptotic Slice Rank and its Connection with Matrix Multiplication

The tensors we study are 3-tensors, which can be seen as trilinear forms over three sets X, Y, Z of formal variables. The slice rank $S(T)$ of a tensor T is a measure of the complexity of T, analogous to the rank of a matrix. In this paper we study the *asymptotic slice rank* $\tilde{S}(T)$ of tensors T:

$$\tilde{S}(T) := \limsup_{n \in \mathbb{N}} S(T^{\otimes n})^{1/n}.$$

$\tilde{S}$ satisfies two key properties:

1. Degenerations cannot increase the asymptotic slice rank of a tensor. In other words, if A degenerates to B, then $\tilde{S}(B) \leq \tilde{S}(A)$.
2. Matrix multiplication tensors have high asymptotic slice rank.

This means that if a certain tensor T has a small value of $\omega_u(T)$, or in other words, powers $T^{\otimes n}$ can degenerate into large matrix multiplication tensors, then T itself must have large asymptotic slice rank. Hence, in order to lower bound $\omega_u(T)$, it suffices to upper bound $\tilde{S}(T)$.

Section 4: Tools for Upper Bounding Asymptotic Slice Rank

In general, bounding $\tilde{S}(T)$ for a tensor T can be much more difficult than bounding $S(T)$. This is because S can be supermultiplicative, i.e. there are tensors A and B such that $S(A) \cdot S(B) \ll S(A \otimes B)$. Indeed, we will show that $\tilde{S}(T) > S(T)$ for many tensors T of interest, including $T = CW_q$.

We will give three new tools for upper bounding $\tilde{S}(T)$ for many tensors T. Each applies to tensors with different properties:

- Theorem 16: If T is over X, Y, Z, then it is straightforward to see that if one of the variable sets is not too large, then $\tilde{S}(T)$ must be small: $\tilde{S}(T) \leq \min\{|X|, |Y|, |Z|\}$. In this first tool, we show how if T can be written as a sum $T = T_1 + \cdots + T_k$ of a few tensors, and each T_i does not have many of one type of variable, then we can still derive an upper bound on $\tilde{S}(T)$.
- Theorem 18: The second tool concerns partitions of the variable sets X, Y, Z. It shows that if $\tilde{S}(T)$ is large, then there is a probability distribution on the blocks of T (subtensors corresponding to a choice of one part from each of the three partitions) so that the total probability mass assigned to each part of each partition is proportional to its size. Loosely, this means that T must have many different "symmetries", no matter how its variables are partitioned.
- Theorem 22: Typically, for tensors A and B, even if $\tilde{S}(A)$ and $\tilde{S}(B)$ are "small", it may still be the case that $\tilde{S}(A + B)$ is large. This third tool shows that if A has an additional property, then one can still bound $\tilde{S}(A + B)$. Roughly, the property that A must satisfy is that not only is $\tilde{S}(A)$ small, but a related notion called the "x-rank" of A must also be small.

In particular, we will remark that our three tools for bounding $\tilde{S}(T)$ strictly generalize similar tools introduced by [2] for bounding $\tilde{I}(T)$. Hence, we generalize their results bounding $\omega_g(T)$ for various tensors T to bounds on $\omega_u(T)$.

Section 5: Universal Method Lower Bounds

We apply our tools to prove upper bounds on $\tilde{S}(T)$, and hence lower bounds on $\omega_u(T)$, for a number of tensors T of interest. To prove Theorem 3, we show that *all three* tools can be applied to CW_q. We also apply our tools to many other tensors of interest including the generalized Coppersmith-Winograd tensors $CW_{q,\sigma}$, the generalized small Coppersmith-Winograd tensors $cw_{q,\sigma}$, the structural tensor T_q of the cyclic group C_q as well as its "lower triangular version" T_q^{lower} , and the subtensor t_{112} of $CW_q^{\otimes 2}$ which arises in [12, 13, 26, 18, 17, 14]. Throughout Section 5 we give many tables of concrete lower bounds that we prove for the tensors in all these different families.

Section 6: "Completeness" of the Laser Method

Finally, we study the Laser Method. The Laser Method applied to a tensor T shows that powers $T^{\otimes n}$ can zero out into large matrix multiplication tensors. Using the properties of $\tilde{S}$ that we prove in Section 3.6, we will show that the Laser Method can also be applied to a tensor T to prove a lower bound on $\tilde{S}(T)$. (More precisely, it actually proves a lower bound on $\tilde{Q}(T)$, the *asymptotic subrank of* T, which in turn lower bounds $\tilde{S}(T)$).

We prove Theorem 4 by combining this construction with Theorem 18, one of our tools for upper bounding $\tilde{S}(T)$. Intuitively, both Theorem 18 and the Laser Method are concerned with probability distributions on blocks of a tensor, and both involve counting the number of variables in powers $T^{\otimes n}$ that are consistent with these distributions. We use this intuition to show that the upper bound given by Theorem 18 is equal to the lower bound given by the Laser Method.

3 Preliminaries

We begin by introducing the relevant notions and notation related to tensors and matrix multiplication. We will use the same notation introduced in [2, Section 3], and readers familiar with that paper may skip to Subsection 3.5.

3.1 Tensor Basics

For sets $X = \{x_1, \ldots, x_q\}$, $Y = \{y_1, \ldots, y_r\}$, and $Z = \{z_1, \ldots, z_s\}$ of formal variables, a *tensor over* X, Y, Z is a trilinear form

$$T = \sum_{x_i \in X, y_j \in Y, z_k \in Z} \alpha_{ijk} x_i y_j z_k,$$

where the α_{ijk} coefficients come from an underlying field $\mathbb{F}$. The *terms*, which we write as $x_i y_j z_k$, are sometimes written as $x_i \otimes y_j \otimes z_k$ in the literature. We say T is *minimal for* X, Y, Z if, for each $x_i \in X$, there is a term involving x_i with a nonzero coefficient in T, and similarly for Y and Z (i.e. T can't be seen as a tensor over a strict subset of the variables). We say that two tensors T_1, T_2 are *isomorphic*, written $T_1 \simeq T_2$, if they are equal up to renaming variables.

If T_1 is a tensor over X_1, Y_1, Z_1, and T_2 is a tensor over X_2, Y_2, Z_2, then the *tensor product* $T_1 \otimes T_2$ is a tensor over $X_1 \times X_2, Y_1 \times Y_2, Z_1 \times Z_2$ such that, for any $(x_1, x_2) \in X_1 \times X_2$, $(y_1, y_2) \in Y_1 \times Y_2$, and $(z_1, z_2) \in Z_1 \times Z_2$, the coefficient of $(x_1, x_2)(y_1, y_2)(z_1, z_2)$ in $T_1 \otimes T_2$ is the product of the coefficient of $x_1 y_1 z_1$ in T_1, and the coefficient of $x_2 y_2 z_2$ in T_2. For any tensor T and positive integer n, the tensor power $T^{\otimes n}$ is the tensor over X^n, Y^n, Z^n resulting from taking the tensor product of n copies of T.

If T_1 is a tensor over X_1, Y_1, Z_1, and T_2 is a tensor over X_2, Y_2, Z_2, then the *direct sum* $T_1 \oplus T_2$ is a tensor over $X_1 \sqcup X_2$, $Y_1 \sqcup Y_2$, $Z_1 \sqcup Z_2$ which results from forcing the variable sets to be disjoint (as in a normal disjoint union) and then summing the two tensors. For a nonnegative integer m and tensor T we write $m \odot T$ for the disjoint sum of m copies of T.

3.2 Tensor Rank

A tensor T has *rank one* if there are values $a_i \in \mathbb{F}$ for each $x_i \in X$, $b_j \in \mathbb{F}$ for each $y_j \in Y$, and $c_k \in \mathbb{F}$ for each $z_k \in Z$, such that the coefficient of $x_i y_j z_k$ in T is $a_i b_j c_k$, or in other words,

$$T = \sum_{x_i \in X, y_j \in Y, z_k \in Z} a_i b_j c_k \cdot x_i y_j z_k = \left(\sum_{x_i \in X} a_i x_i \right) \left(\sum_{y_j \in Y} b_j y_j \right) \left(\sum_{z_k \in Z} c_k z_k \right).$$

The *rank* of a tensor T, denoted $R(T)$, is the smallest number of rank one tensors whose sum (summing the coefficient of each term individually) is T. It is not hard to see that for tensors T and positive integers n, we always have $R(T^{\otimes n}) \leq R(T)^n$, but for some tensors T of interest this inequality is not tight. We thus define the *asymptotic rank* of tensor T as $\tilde{R}(T) := \liminf_{n \in \mathbb{N}} (R(T^{\otimes n}))^{1/n}$.

3.3 Matrix Multiplication Tensors

For positive integers a, b, c, the matrix multiplication tensor $\langle a, b, c \rangle$ is a tensor over $\{x_{ij}\}_{i \in [a], j \in [b]}$, $\{y_{jk}\}_{j \in [b], k \in [c]}$, $\{z_{ki}\}_{k \in [c], i \in [a]}$ given by

$$\langle a, b, c \rangle = \sum_{i=1}^{a} \sum_{j=1}^{b} \sum_{k=1}^{c} x_{ij} y_{jk} z_{ki}.$$

It is not hard to verify that for positive integers $a_1, a_2, b_1, b_2, c_1, c_2$, we have $\langle a_1, b_1, c_1 \rangle \otimes \langle a_2, b_2, c_2 \rangle \simeq \langle a_1 a_2, b_1 b_2, c_1 c_2 \rangle$. The *exponent of matrix multiplication*, denoted ω, is defined as

$$\omega := \liminf_{a,b,c \in \mathbb{N}} 3 \log_{abc}(R(\langle a, b, c \rangle)).$$

Because of the tensor product property above, we can alternatively define ω in a number of ways:

$$\omega = \liminf_{a,b,c\in\mathbb{N}} 3\log_{abc}(\tilde{R}(\langle a,b,c\rangle)) = \liminf_{n\in\mathbb{N}} \log_n \tilde{R}(\langle n,n,n\rangle) = \log_2(\tilde{R}(\langle 2,2,2\rangle)).$$

For instance, Strassen [21] showed that $R(\langle 2,2,2\rangle) \leq 7$, which implies that $\omega \leq \log_2(7)$.

3.4 Degenerations and the Universal Method

We now describe a very general way to transform from a tensor T_1 over X_1, Y_1, Z_1 to a tensor T_2 over X_2, Y_2, Z_2. For a formal variable λ, pick maps $\alpha : X_1 \times X_2 \to \mathbb{F}(\lambda)$, $\beta : Y_1 \times Y_2 \to \mathbb{F}(\lambda)$, and $\gamma : Z_1 \times Z_2 \to \mathbb{F}(\lambda)$, which map pairs of variables to polynomials in λ, and pick an integer h. Then, when you replace each $x \in X_1$ with $\sum_{x'\in X_2} \alpha(x,x')x'$, each $y \in Y_1$ with $\sum_{y'\in Y_2} \beta(y,y')y'$, and each $z \in Z_1$ with $\sum_{z'\in Z_2} \gamma(z,z')z'$, in T_1, then the resulting tensor T' is a tensor over X_2, Y_2, Z_2 with coefficients over $\mathbb{F}(\lambda)$. When T' is instead viewed as a polynomial in λ whose coefficients are tensors over X_2, Y_2, Z_2 with coefficients in $\mathbb{F}$, it must be that T_2 is the coefficient of λ^h, and the coefficient of $\lambda^{h'}$ is 0 for all $h' < h$.

If such a transformation is possible, we say T_2 is a *degeneration* of T_1. There are also two more restrictive types of degenerations:

- T_2 is a *monomial degeneration* of T_1 if such a transformation is possible where the polynomials in the ranges of α, β, γ have at most one monomial, and furthermore, for each $x \in X_1$ there is at most one $x' \in X_2$ such that $\alpha(x,x') \neq 0$, and similarly for β and γ.[3]
- T_2 is a *zeroing out* of T_1 if, in addition to the restrictions of a monomial degeneration, the ranges of α, β, γ must be $\{0,1\}$.

Degenerations are useful in the context of matrix multiplication algorithms because degenerations cannot increase the rank of a tensor. In other words, if T_2 is a degeneration of T_1, then $R(T_2) \leq R(T_1)$ [4]. It is often hard to bound the rank of matrix multiplication tensors directly, so all known approaches proceed by bounding the rank of a different tensor T and then showing that powers of T degenerate into matrix multiplication tensors.

More precisely, all known approaches fall within the following method, which we call the *Universal Method* [2] applied to a tensor T of asymptotic rank $R = \tilde{R}(T)$: Consider all positive integers n, and all ways to degenerate $T^{\otimes n}$ into a disjoint sum $\bigoplus_{i=1}^{m} \langle a_i, b_i, c_i\rangle$ of matrix multiplication tensors, resulting in an upper bound on ω by the asymptotic sum inequality [19] of $\sum_{i=1}^{m} (a_i b_i c_i)^{\omega/3} \leq R^n$. Then, $\omega_u(T)$, the bound on ω from the Universal Method applied to T, is the lim inf over all such n and degenerations, of the resulting upper bound on ω.

In [2], two weaker versions of the Universal Method are also defined: *the Galactic Method*, in which the degeneration must be a monomial degeneration, resulting in a bound $\omega_g(T)$, and *the Solar Method*, in which the degeneration must be a zeroing out, resulting in a bound $\omega_s(T)$. To be clear, all three of these methods are very general, and we don't know the values of $\omega_s(T)$, $\omega_g(T)$, or $\omega_u(T)$ for almost any nontrivial tensors T. In fact, all the known

[3] Some definitions of monomial degenerations do not have this second condition, or equivalently, consider a monomial degeneration to be a "restriction" composed with what we defined here. The distinction is not important for this paper, but we give this definition since it captures Strassen's monomial degeneration from matrix multiplication tensors to independent tensors [22] (see also Proposition 10 below), and it is the notion that the prior work [2] proved lower bounds against.

approaches to bounding ω proceed by giving upper bounds on $\omega_s(T)$ for some carefully chosen tensors T; the most successful has been the Coppersmith-Winograd family of tensors $T = CW_q$, which has yielded all the best known bounds on ω since the 80's [11, 13, 26, 18]. Indeed, the two most successful approaches, the Laser Method [20] and the Group-Theoretic Approach [9] ultimately use zeroing outs of tensors. We refer the reader to [2, Sections 3.3 and 3.4] for more details on these approaches and how they relate to the notions used here.

3.5 Tensor Value

Coppersmith and Winograd [12] defined the *value* of a tensor in their analysis of the CW_q tensor. For a tensor T, and any $\tau \in [2/3, 1]$, the τ-value of T, denoted $V_\tau(T)$, is defined as follows: Consider all positive integers n, and all ways σ to degenerate $T^{\otimes n}$ into a direct sum $\bigoplus_{i=1}^{q(\sigma)} \langle a_i^\sigma, b_i^\sigma, c_i^\sigma \rangle$ of matrix multiplication tensors. Then, $V_\tau(T)$ is given by

$$V_\tau(T) := \limsup_{n,\sigma} \left(\sum_{i=1}^{q(\sigma)} (a_i^\sigma b_i^\sigma c_i^\sigma)^\tau \right)^{1/n}.$$

We can then equivalently define $\omega_u(T)$ as the lim inf of ω_u, over all $\omega_u \in [2, 3]$ such that $V_{\omega_u/3}(T) \geq \tilde{R}(T)$. We can see from the power mean inequality that $V_\tau(T) \geq V_{2/3}(T)^{3\tau/2}$ for all $\tau \in [2/3, 1]$, although this bound is often not tight as there can be better degenerations of $T^{\otimes n}$ depending on the value of τ.

3.6 Asymptotic Slice Rank

The main new notions we will need in this paper relate to the slice rank of tensors. We say a tensor T over X, Y, Z has *x-rank* 1 if it is of the form

$$T = \left(\sum_{x \in X} \alpha_x \cdot x \right) \otimes \left(\sum_{y \in Y} \sum_{z \in Z} \beta_{y,z} \cdot y \otimes z \right) = \sum_{x \in X, y \in Y, z \in Z} \alpha_x \beta_{y,z} \cdot xyz$$

for some choices of the α and β coefficients over the base field. More generally, the x-rank of T, denoted $\mathrm{S_x}(T)$, is the minimum number of tensors of x-rank 1 whose sum is T. We can similarly define the y-rank, $\mathrm{S_y}$, and the z-rank, $\mathrm{S_z}$. Then, the *slice rank* of T, denoted $\mathrm{S}(T)$, is the minimum k such that there are tensors T_X, T_Y and T_Z with $T = T_X + T_Y + T_Z$ and $\mathrm{S_x}(T_X) + \mathrm{S_y}(T_Y) + \mathrm{S_z}(T_Z) = k$.

Unlike tensor rank, the slice-rank is not submultiplicative in general, i.e. there are tensors A and B such that $\mathrm{S}(A \otimes B) > \mathrm{S}(A) \cdot \mathrm{S}(B)$. For instance, it is not hard to see that $\mathrm{S}(CW_5) = 3$, but since it is known [26, 18] that $\omega_s(CW_5) \leq 2.373$, it follows (e.g. from Theorem 14 below) that $\mathrm{S}(CW_q^{\otimes n}) \geq 7^{n \cdot 2/2.373 - o(n)} \geq 5.15^{n - o(n)}$. We are thus interested in the *asymptotic slice rank*, $\tilde{S}(T)$, of tensors T, defined as

$$\tilde{S}(T) := \limsup_{n \in \mathbb{N}} [\mathrm{S}(T^{\otimes n})]^{1/n}.$$

We note a few simple properties of slice rank which will be helpful in our proofs:

▶ **Lemma 6.** *For tensors A and B:*

(1) $\mathrm{S}(A) \leq \mathrm{S_x}(A) \leq \mathrm{R}(A)$,
(2) $\mathrm{S_x}(A \otimes B) \leq \mathrm{S_x}(A) \cdot \mathrm{S_x}(B)$,
(3) $\mathrm{S}(A + B) \leq \mathrm{S}(A) + \mathrm{S}(B)$, *and* $\mathrm{S_x}(A + B) \leq \mathrm{S_x}(A) + \mathrm{S_x}(B)$,
(4) $\mathrm{S}(A \otimes B) \leq \mathrm{S}(A) \cdot \max\{\mathrm{S_x}(B), \mathrm{S_y}(B), \mathrm{S_z}(B)\}$, *and*
(5) *If A is a tensor over X, Y, Z, then* $\mathrm{S_x}(T) \leq |X|$ *and hence* $\mathrm{S}(T) \leq \min\{|X|, |Y|, |Z|\}$.

Proof. (1) and (2) are straightforward. (3) follows since the sum of the slice rank (resp. x-rank) expressions for A and for B gives a slice rank (resp. x-rank) expression for $A+B$. To prove (4), let $m = \max\{\mathrm{S_x}(B), \mathrm{S_y}(B), \mathrm{S_z}(B)\}$, and note that if $A = A_X + A_Y + A_Z$ such that $\mathrm{S_x}(A_X) + \mathrm{S_y}(A_Y) + \mathrm{S_z}(A_Z) = \mathrm{S}(A)$, then

$$A \otimes B = A_X \otimes B + A_Y \otimes B + A_Z \otimes B,$$

and so

$$\begin{aligned}\mathrm{S}(A \otimes B) &\leq \mathrm{S}(A_X \otimes B) + \mathrm{S}(A_Y \otimes B) + \mathrm{S}(A_Z \otimes B)\\ &\leq \mathrm{S_x}(A_X \otimes B) + \mathrm{S_y}(A_Y \otimes B) + \mathrm{S_z}(A_Z \otimes B)\\ &\leq \mathrm{S_x}(A_X)\,\mathrm{S_x}(B) + \mathrm{S_y}(A_Y)\,\mathrm{S_y}(B) + \mathrm{S_z}(A_Z)\,\mathrm{S_z}(B)\\ &\leq \mathrm{S_x}(A_X)m + \mathrm{S_y}(A_Y)m + \mathrm{S_z}(A_Z)m = \mathrm{S}(A)\cdot m.\end{aligned}$$

Finally, (5) follows since, for instance, any tensor with one only x-variable has x-rank 1. ◀

Asymptotic slice rank is interesting in the context of matrix multiplication algorithms because of the following facts.

▶ **Definition 7.** *For a positive integer q, the* independent tensor of size q, *denoted $\langle q\rangle$, is the tensor $\sum_{i=1}^{q} x_i y_i z_i$ with q terms that do not share any variables.*

▶ **Proposition 8** ([25] Corollary 2)**.** *If A and B are tensors such that A has a degeneration to B, then $\mathrm{S}(B) \leq \mathrm{S}(A)$, and hence $\tilde{S}(B) \leq \tilde{S}(A)$.*

▶ **Proposition 9** ([24] Lemma 1; see also [5] Lemma 4.7)**.** *For any positive integer q, we have $\mathrm{S}(\langle q\rangle) = \tilde{S}(\langle q\rangle) = q$, where $\langle q\rangle$ is the independent tensor of size q.*

▶ **Proposition 10** ([22] Theorem 4; see also [2] Lemma 4.2)**.** *For any positive integers a, b, c, the matrix multiplication tensor $\langle a, b, c\rangle$ has a (monomial) degeneration to an independent tensor of size at least $0.75 \cdot abc/\max\{a, b, c\}$.*

▶ **Corollary 11.** *For any positive integers a, b, c, we have $\tilde{S}(\langle a, b, c\rangle) = abc/\max\{a, b, c\}$.*

Proof. Assume without loss of generality that $c \geq a, b$. For any positive integer n, we have that $\langle a, b, c\rangle^{\otimes n} \simeq \langle a^n, b^n, c^n\rangle$ has a degeneration to an independent tensor of size at least $0.75 \cdot a^n b^n$, meaning $S(\langle a, b, c\rangle^{\otimes n}) \geq 0.75 \cdot a^n b^n$ and hence $\tilde{S}(\langle a, b, c\rangle) \geq (0.75)^{1/n} ab$, which means $\tilde{S}(\langle a, b, c\rangle) \geq ab$. Meanwhile, $\langle a, b, c\rangle$ has ab different x-variables, so it must have $\mathrm{S_x}(\langle a, b, c\rangle) \leq ab$ and more generally, $\mathrm{S}(\langle a, b, c\rangle^{\otimes n}) \leq \mathrm{S_x}(\langle a, b, c\rangle^{\otimes n}) \leq (ab)^n$, which means $\tilde{S}(\langle a, b, c\rangle) \leq ab$. ◀

To summarize: we know that degenerations cannot increase asymptotic slice rank, and that matrix multiplication tensors have a high asymptotic slice rank. Hence, if T is a tensor such that $\omega_u(T)$ is "small', meaning a power of T has a degeneration to a disjoint sum of many large matrix multiplication tensors, then T itself must have "large" asymptotic slice rank. This can be formalized identically to [2, Theorem 4.1 and Corollary 4.3] to show:

▶ **Theorem 12.** *For any tensor T,*

$$\tilde{S}(T) \geq \tilde{R}(T)^{\frac{6}{\omega_u(T)} - 2}.$$

▶ **Corollary 13.** *For any tensor T, if $\omega_u(T) = 2$, then $\tilde{S}(T) = \tilde{R}(T)$. Moreover, for every constant $s < 1$, there is a constant $w > 2$ such that every tensor T with $\tilde{S}(T) \leq \tilde{R}(T)^s$ must have $\omega_u(T) \geq w$.*

Almost all the tensors we consider in this note are *variable-symmetric* tensors, and for these tensors T we can get a better lower bound on $\omega_u(T)$ from an upper bound on $\tilde{S}(T)$. We say that a tensor T over X, Y, Z is variable-symmetric if $|X| = |Y| = |Z|$, and the coefficient of $x_i y_j z_k$ equals the coefficient of $x_j y_k z_i$ in T for all $(x_i, y_j, z_k) \in X \times Y \times Z$.

▶ **Theorem 14.** *For a variable-symmetric tensor T we have $\omega_u(T) \geq 2\log(\tilde{R}(T))/\log(\tilde{S}(T))$.*

Proof. As in the proof of [2, Theorem 4.1], by definition of ω_u, we know that for every $\delta > 0$, there is a positive integer n such that $T^{\otimes n}$ has a degeneration to $F \odot \langle a, b, c\rangle$ for integers F, a, b, c such that $\omega_u(T)^{1+\delta} \geq 3\log(\tilde{R}(T)^n/F)/\log(abc)$. In fact, since T is symmetric, we know $T^{\otimes n}$ also has a degeneration to $F \odot \langle b, c, a\rangle$ and to $F \odot \langle c, a, b\rangle$, and so $T^{\otimes 3n}$ has a degeneration to $F^3 \odot \langle abc, abc, abc\rangle$. As above, it follows that $\tilde{S}(T^{\otimes 3n}) \geq \tilde{S}(F^3 \odot \langle abc, abc, abc\rangle) = F^3 \cdot (abc)^2$. Rearranging, we see

$$abc \leq \tilde{S}(T)^{3n/2}/F^{3/2}.$$

Hence,

$$\begin{aligned}\omega_u(T)^{1+\delta} &\geq 3\frac{\log(\tilde{R}(T)^n/F)}{\log(abc)} \geq 3\frac{\log(\tilde{R}(T)^n/F)}{\log(\tilde{S}(T)^{3n/2}/F^{3/2})} \\ &= 2\frac{\log(\tilde{R}(T)) - \frac{1}{n}\log(F)}{\log(\tilde{S}(T)) - \frac{1}{n}\log(F)} \geq 2\frac{\log(\tilde{R}(T))}{\log(\tilde{S}(T))},\end{aligned}$$

where the last step follows because $\tilde{R}(T) \geq \tilde{S}(T)$ and so subtracting the same quantity from both the numerator and denominator cannot decrease the value of the fraction. This holds for all $\delta > 0$ and hence implies the desired result. ◀

Slice Rank versus Subrank

For a tensor T, let $Q'(T)$ denote the largest integer q such that there is a degeneration from T to $\langle q\rangle$. The *asymptotic subrank* of T is defined as $\tilde{Q}(T) := \limsup_{n\in\mathbb{N}} Q'(T^{\otimes n})^{1/n}$. Propositions 8 and 9 above imply that $\tilde{Q}(T) \leq \tilde{S}(T)$ for all tensors T. Similarly, it is not hard to see that Theorems 12 and 14 hold with $\tilde{S}$ replaced by $\tilde{Q}$. One could thus conceivably hope to prove stronger lower bounds than those in this paper by bounding $\tilde{Q}$ instead of $\tilde{S}$. However, we will prove in Corollary 29 below that $\tilde{Q}(T) = \tilde{S}(T)$ for every tensor we study in this paper, so such an improvement using $\tilde{Q}$ is impossible. More generally, there are currently no known tensors T for which the best known upper bound on $\tilde{Q}(T)$ is smaller than the best known upper bound on $\tilde{S}(T)$ (including the new bounds of [7, 6]). Hence, novel tools for upper bounding $\tilde{Q}$ would be required for such an approach to proving better lower bounds on ω_u.

3.7 Partition Notation

In a number of our results, we will be partitioning the terms of tensors into blocks defined by partitions of the three variable sets. Here we introduce some notation for some properties of such partitions; these definitions all depend on the particular partition of the variables being used, which will be clear from context.

Suppose T is a tensor minimal over X, Y, Z, and let $X = X_1 \cup \cdots \cup X_{k_X}$, $Y = Y_1 \cup \cdots \cup Y_{k_Y}$, $Z = Z_1 \cup \cdots \cup Z_{k_Z}$ be partitions of the three variable sets. For $(i, j, k) \in [k_X] \times [k_Y] \times [k_Z]$, let T_{ijk} be T restricted to X_i, Y_j, Z_k (i.e. T with $X \setminus X_i$, $Y \setminus Y_j$, and $Z \setminus Z_k$ zeroed out),

and let $L = \{T_{ijk} \mid (i,j,k) \in [k_X] \times [k_Y] \times [k_Z], T_{ijk} \neq 0\}$. T_{ijk} is called a *block* of T. For $i \in [k_X]$ let $L_{X_i} = \{T_{ij'k'} \in L \mid (j',k') \in [k_Y] \times [k_Z]\}$, and define similarly L_{Y_j} and L_{Z_k}.

We will be particularly interested in probability distributions $p : L \to [0,1]$. Let $P(L)$ be the set of such distributions. For such a $p \in P(L)$, and for $i \in [k_X]$, let $p(X_i) := \sum_{T_{ijk} \in L_{X_i}} p(T_{ijk})$, and similarly $p(Y_j)$ and $p(Z_k)$. Then, define $p_X \in \mathbb{R}$ by

$$p_X := \prod_{i \in [k_X]} \left(\frac{|X_i|}{p(X_i)} \right)^{p(X_i)},$$

and p_Y and p_Z similarly. This expression, which arises naturally in the Laser Method, will play an important role in our upper bounds and lower bounds.

3.8 Tensor Rotations and Variable-Symmetric Tensors

If T is a tensor over X, Y, Z, then the *rotation of* T, denoted $rot(T)$, is the tensor over Y, Z, X such that for any $(x_i, y_j, z_k) \in X \times Y \times Z$, the coefficient of $x_i y_j z_k$ in T is equal to the coefficient of $y_j z_k x_i$ in $rot(T)$. Tensor T is *variable-symmetric* if $T \simeq rot(T)$.

If T is a variable-symmetric tensor minimal over X, Y, Z, then partitions $X = X_1 \cup \cdots \cup X_{k_X}$, $Y = Y_1 \cup \cdots \cup Y_{k_Y}$, $Z = Z_1 \cup \cdots \cup Z_{k_Z}$ of the variable sets are called *T-symmetric* if (using the notation of the previous subsection) $k_X = k_Y = k_Z$, $|X_i| = |Y_i| = |Z_i|$ for all $i \in [k_X]$, and the block $T_{jki} \simeq rot(T_{ijk})$ for all $(i,j,k) \in [k_X]^3$. For the L resulting from such a T-symmetric partition, a probability distribution $p \in P(L)$ is called *T-symmetric* if it satisfies $p(T_{ijk}) = p(T_{jki})$ for all $(i,j,k) \in [k_X]^3$, and we write $P^{sym}(L) \subseteq P(L)$ for the set of such T-symmetric distributions. Notice in particular that any $p \in P^{sym}(L)$ satisfies $p_X = p_Y = p_Z$.

4 Combinatorial Tools for Asymptotic Slice Rank Upper Bounds

We now give three general tools for proving upper bounds on $\tilde{S}(T)$ for many tensors T. Each of our tools generalizes one of the three main tools of [2], which were bounding the weaker notion $\tilde{I}$ instead of $\tilde{S}$, and could also only apply to a more restrictive set of tensors. We will make clear what previous result we are generalizing, although our presentation here is entirely self-contained.

4.1 Generalization of [2, Theorem 5.3]

We know that tensors T without many of one variable have small $\tilde{S}(T)$. We begin by showing that if T can be written as a sum of a few tensors, each of which does not have many of one variable, then we can still prove an upper bound on $\tilde{S}(T)$.

If X, Y, Z are minimal for T, then the *measure* of T, denoted $\mu(T)$, is given by $\mu(T) := |X| \cdot |Y| \cdot |Z|$. We state two simple facts about μ:

▶ **Fact 15.** *For tensors A and B,*

- $\mu(A \otimes B) = \mu(A) \cdot \mu(B)$, *and*
- *if A is minimal over X, Y, Z, then* $\mathrm{S}(A) \leq \min\{|X|, |Y|, |Z|\} \leq \mu(A)^{1/3}$.

▶ **Theorem 16.** *Suppose T is a tensor, and $T_1, \ldots, T_k$ are tensors with $T = T_1 + \cdots + T_k$. Then, $\tilde{S}(T) \leq \sum_{i=1}^{k} (\mu(T_i))^{1/3}$.*

Proof. Note that

$$T^{\otimes n} = \sum_{(P_1,\ldots,P_n)\in\{T_1,\ldots,T_k\}^n} P_1 \otimes \cdots \otimes P_n.$$

It follows that

$$\begin{aligned}
\mathrm{S}(T^{\otimes n}) &\leq \sum_{(P_1,\ldots,P_n)\in\{T_1,\ldots,T_k\}^n} \mathrm{S}(P_1 \otimes \cdots \otimes P_n) \\
&\leq \sum_{(P_1,\ldots,P_n)\in\{T_1,\ldots,T_k\}^n} \mu(P_1 \otimes \cdots \otimes P_n)^{1/3} \\
&= \sum_{(P_1,\ldots,P_n)\in\{T_1,\ldots,T_k\}^n} (\mu(P_1)\cdot\mu(P_2)\cdots\mu(P_n))^{1/3} \\
&= (\mu(T_1)^{1/3} + \cdots + \mu(T_k)^{1/3})^n,
\end{aligned}$$

which implies as desired that $\tilde{S}(T) \leq (\mu(T_1)^{1/3} + \cdots + \mu(T_k)^{1/3})$. ◀

▶ Remark 17. [2, Theorem 5.3], in addition to bounding $\tilde{I}$ instead of $\tilde{S}$, also required that $T = T_1 + \cdots + T_k$ be a *partition* of the terms of T. Here in Theorem 16 we are allowed any tensor sum, although in general a partition minimizes the resulting upper bound.

4.2 Generalization of [2, Theorem 5.2]

This tool will be the most important in upper bounding the asymptotic slice rank of many tensors of interest. We show that a partitioning method similar to the Laser Method applied to a tensor T can be used to prove upper bounds on $\tilde{S}(T)$. Recall the definitions and notation about partitions of tensors from Section 3.7.

▶ **Theorem 18.** *For any tensor T and partition of its variable sets,*

$$\tilde{S}(T) \leq \limsup_{p\in P(L)} \min\{p_X, p_Y, p_Z\}.$$

Proof. For any positive integer n, we can write

$$T^{\otimes n} = \sum_{(P_1,\ldots,P_n)\in L^n} P_1 \otimes \cdots \otimes P_n.$$

For a given $(P_1,\ldots,P_n) \in L^n$, let $dist(P_1,\ldots,P_n)$ be the probability distribution on L which results from picking a uniformly random $\alpha \in [n]$ and outputting P_α. For a probability distribution $p : L \to [0,1]$, define $L_{n,p} := \{(P_1,\ldots,P_n) \in L^n \mid dist(P_1,\ldots,P_n) = p\}$. Note that the number of p for which $L_{n,p}$ is nonempty is only $\mathrm{poly}(n)$, since they are the distributions which assign an integer multiple of $1/n$ to each element of L. Let D be the set of these probability distributions.

We can now rearrange:

$$T^{\otimes n} = \sum_{p\in D} \sum_{(P_1,\ldots,P_n)\in L_{n,p}} P_1 \otimes \cdots \otimes P_n.$$

Hence,

$$\begin{aligned}
\mathrm{S}(T^{\otimes n}) &\leq \sum_{p\in D} \mathrm{S}\left(\sum_{(P_1,\ldots,P_n)\in L_{n,p}} P_1 \otimes \cdots \otimes P_n \right) \\
&\leq \mathrm{poly}(n) \cdot \max_{p\in D} \mathrm{S}\left(\sum_{(P_1,\ldots,P_n)\in L_{n,p}} P_1 \otimes \cdots \otimes P_n \right).
\end{aligned}$$

For any probability distribution $p : L \to [0,1]$, let us count the number of x-variables used in $\left(\sum_{(P_1,\ldots,P_n)\in L_{n,p}} P_1 \otimes \cdots \otimes P_n\right)$. These are the tuples of the form $(x_1,\ldots,x_n) \in X^n$ where, for each $i \in [k_X]$, there are exactly $n \cdot p(X_i)$ choices of j for which $x_j \in X_i$. The number of these is[4]

$$\binom{n}{n\cdot p(X_1), n\cdot p(X_2), \ldots, n\cdot p(X_{k_X})} \cdot \prod_{i\in[k_X]} |X_i|^{n\cdot p(X_i)}.$$

This is upper bounded by $p_X^{n+o(n)}$, where p_X is the quantity defined in Section 3.7. It follows that $\mathrm{S_x}\left(\sum_{(P_1,\ldots,P_n)\in L_{n,p}} P_1 \otimes \cdots \otimes P_n\right) \leq p_X^{n+o(n)}$. We can similarly argue about $\mathrm{S_y}$ and $\mathrm{S_z}$. Hence,

$$\begin{aligned}
\mathrm{S}(T^{\otimes n}) &\leq \mathrm{poly}(n) \cdot \max_{p\in D} \mathrm{S}\left(\sum_{(P_1,\ldots,P_n)\in L_{n,p}} P_1 \otimes \cdots \otimes P_n\right) \\
&\leq \mathrm{poly}(n) \cdot \max_{p\in D} \min\{p_X, p_Y, p_Z\}^{n+o(n)} \\
&\leq \mathrm{poly}(n) \cdot \limsup_{p\in P(L)} \min\{p_X, p_Y, p_Z\}^{n+o(n)}.
\end{aligned}$$

Hence, $\mathrm{S}(T^{\otimes n}) \leq \limsup_p \min\{p_X, p_Y, p_Z\}^{n+o(n)}$, and the desired result follows. ◀

▶ Remark 19. [2, Theorem 5.2] is less general than our Theorem 18 in two ways: it used $\tilde{I}$ instead of $\tilde{S}$, and it required each X_i, Y_j, Z_k to contain only one variable.

▶ Remark 20. Suppose T is over X, Y, Z with $|X| = |Y| = |Z| = q$. For any probability distribution p we always have $p_X, p_Y, p_Z \leq q$, and moreover we only have $p_X = q$ when $p(X_i) = |X_i|/q$ for each i. Similar to [2, Corollary 5.1], it follows that if no probability distribution p is δ-close (say, in ℓ_1 distance) to having $p(X_i) = |X_i|/q$ for all i, $p(Y_j) = |Y_j|/q$ for all j, and $p(Z_k) = |Z_k|/q$ for all k, simultaneously, then we get $\tilde{S}(T) \leq q^{1-f(\delta)}$ for some increasing function f with $f(\delta) > 0$ for all $\delta > 0$.

We make a remark about applying Theorem 18 to variable-symmetric tensors. This remark has implicitly been used in past work on applying the Laser method, such as [12], but we prove it here for completeness. Recall the notation in Section 3.8 about such tensors.

▶ **Proposition 21.** *Suppose T is a variable-symmetric tensor over X, Y, Z, and $X = X_1 \cup \cdots \cup X_{k_X}$, $Y = Y_1 \cup \cdots \cup Y_{k_Y}$, $Z = Z_1 \cup \cdots \cup Z_{k_Z}$ are T-symmetric partitions. Then,*

$$\tilde{S}(T) \leq \limsup_{p\in P^{sym}(L)} p_X.$$

Proof. We know from Theorem 18 that $\tilde{S}(T) \leq \limsup_{p\in P(L)} \min\{p_X, p_Y, p_Z\}$. We will show that for any $p \in P(L)$, there is a $p' \in P^{sym}(L)$ such that $\min\{p_X, p_Y, p_Z\} \leq \min\{p'_X, p'_Y, p'_Z\}$, which means that in fact, $\tilde{S}(T) \leq \limsup_{p\in P^{sym}(L)} \min\{p_X, p_Y, p_Z\}$. Finally, the desired result will follow since, for any $p' \in P^{sym}(L)$, we have $p'_X = p'_Y = p'_Z$.

[4] Here, $\binom{n}{p_1 n, p_2 n, \ldots, p_\ell n} = \frac{n!}{(p_1 n)!(p_2 n)!\cdots(p_\ell n)!}$, with each $p_i \in [0,1]$ and $p_1 + \cdots + p_\ell = 1$, is the multinomial coefficient, with the known bound from Stirling's approximation, for fixed p_is, that $\binom{n}{p_1 n, p_2 n, \ldots, p_\ell n} \leq \left(\prod_i p_i^{-p_i}\right)^{n+o(n)}$. Throughout this paper we use the convention that $p_i^{p_i} = 1$ when $p_i = 0$.

Consider any $p \in P(L)$, and define the distribution $p' \in P^{sym}(L)$ by $p'(T_{ijk}) := (p(T_{ijk}) + p(T_{jki}) + p(T_{kij}))/3$ for each $T_{ijk} \in L$. In order to show that $\min\{p_X, p_Y, p_Z\} \leq p'_X$, we will show that $(p_X p_Y p_Z)^{1/3} \leq p'_X$:

$$\begin{aligned}
(p_X p_Y p_Z)^{1/3} &= \prod_{i \in [k_X]} \left(\frac{|X_i|}{p(X_i)}\right)^{p(X_i)/3} \left(\frac{|Y_i|}{p(Y_i)}\right)^{p(Y_i)/3} \left(\frac{|Z_i|}{p(Z_i)}\right)^{p(Z_i)/3} \\
&= \prod_{i \in [k_X]} \frac{|X_i|^{p'(X_i)}}{(p(X_i)^{p(X_i)} p(Y_i)^{p(Y_i)} p(Z_i)^{p(Z_i)})^{1/3}} \\
&\leq \prod_{i \in [k_X]} \frac{|X_i|^{p'(X_i)}}{p'(X_i)^{p'(X_i)}} \\
&= p'_X,
\end{aligned}$$

where the second-to-last step follows from the fact that for any real numbers $a, b, c \in [0, 1]$, setting $d = (a + b + c)/3$, we have $a^a b^b c^c \geq d^{3d}$. ◀

4.3 Generalization of [2, Theorem 5.1]

The final remaining tool from [2], their Theorem 5.1, turns out to be unnecessary for proving our tight lower bounds in the next section. Nonetheless, we sketch here how to extend it to give asymptotic slice rank upper bounds as well.

For a tensor T, let $m(T) := \max\{\mathrm{S_x}(T), \mathrm{S_y}(T), \mathrm{S_z}(T)\}$. Recall from Lemma 6 that for any two tensors A, B we have $\mathrm{S}(A \otimes B) \leq \mathrm{S}(A) \cdot m(B)$.

In general, for two tensors A and B, even if $\tilde{S}(A)$ and $\tilde{S}(B)$ are "small", it might still be the case that $\tilde{S}(A+B)$ is "large", much larger than $\tilde{S}(A)+\tilde{S}(B)$. For instance, for any positive integer q, define the tensors $T_1 := \sum_{i=0}^{q} x_0 y_i z_i$, $T_2 := \sum_{i=1}^{q+1} x_i y_0 z_i$, and $T_3 := \sum_{i=1}^{q+1} x_i y_i z_{q+1}$. We can see that $\tilde{S}(T_1) = \tilde{S}(T_2) = \tilde{S}(T_3) = 1$, but $T_1 + T_2 + T_3 = CW_q$, and we will show soon that $\tilde{S}(CW_q)$ grow unboundedly with q.

Here we show that if, not only is $\tilde{S}(A)$ small, but even $\mathrm{S_x}(A)$ is small, then we can get a decent bound on $\tilde{S}(A + B)$.

▶ **Theorem 22.** *Suppose T, A, B are tensors such that $A + B = T$. Then,*

$$\tilde{S}(T) \leq \left(\frac{m(A)}{(1-p) \cdot \mathrm{S_x}(A)}\right)^{1-p} \cdot \frac{1}{p^p},$$

where $p \in [0, 1]$ is given by

$$p := \frac{\log\left(\frac{\mathrm{S_x}(B)}{\tilde{S}(B)}\right)}{\log\left(\frac{m(A)}{\mathrm{S_x}(A)}\right) + \log\left(\frac{\mathrm{S_x}(B)}{\tilde{S}(B)}\right)}.$$

Proof. We begin by, for any integers $n \geq k \geq 0$, giving bounds on $\mathrm{S}(A^{\otimes k} \otimes B^{\otimes(n-k)})$. First, since $\mathrm{S_x}$ is submultiplicative, we have

$$\mathrm{S}(A^{\otimes k} \otimes B^{\otimes(n-k)}) \leq \mathrm{S_x}(A^{\otimes k} \otimes B^{\otimes(n-k)}) \leq \mathrm{S_x}(A)^k \cdot \mathrm{S_x}(B)^{n-k}.$$

Second, from the definition of m, we have

$$\mathrm{S}(A^{\otimes k} \otimes B^{\otimes(n-k)}) \leq m(A^{\otimes k}) \cdot \mathrm{S}(B^{\otimes(n-k)}) \leq m(A)^k \cdot \tilde{S}(B)^{n-k}.$$

It follows that for any positive integer n we have

$$S(T^{\otimes n}) \leq \sum_{k=0}^{n} \binom{n}{k} \cdot \mathrm{S}(A^{\otimes k} \otimes B^{\otimes(n-k)}) \leq \sum_{k=0}^{n} \binom{n}{k} \cdot \min\{\mathrm{S}_{\mathrm{x}}(A)^k \cdot \mathrm{S}_{\mathrm{x}}(B)^{n-k}, m(A)^k \cdot \tilde{S}(B)^{n-k}\}.$$

As in the proof of [2, Theorem 5.1], we can see that the quantity $\binom{n}{k} \cdot \min\{\mathrm{S}_{\mathrm{x}}(A)^k \cdot \mathrm{S}_{\mathrm{x}}(B)^{n-k}, m(A)^k \cdot \tilde{S}(B)^{n-k}\}$ is maximized at $k = pn$, and the result follows. ◀

▶ Remark 23. This result generalizes [2, Theorem 5.1], no longer requiring that A be the tensor T restricted to a single x-variable. In [2, Theorem 5.1], since A is T restricted to a single x-variable, and we required A to have at most q terms, we got the bounds $\mathrm{S}_{\mathrm{x}}(A) = 1$ and $m(A) \leq q$. Similarly, B had at most $q-1$ different x-variables, so $\mathrm{S}_{\mathrm{x}}(B) \leq q-1$. Substituting those values into Theorem 22 yields the original [2, Theorem 5.1] with $\tilde{I}$ replaced by $\tilde{S}$.

5 Computing the Slice Ranks for Tensors of Interest

In this section, we give slice rank upper bounds for a number of tensors of interest. It will follow from Section 6 that *all of the bounds we prove in this Section are tight.*

5.1 Generalized Coppersmith-Winograd Tensors

We begin with the generalized CW tensors defined in [2], which for a positive integer q and a permutation $\sigma : [q] \to [q]$ are given by

$$CW_{q,\sigma} := x_0 y_0 z_{q+1} + x_0 y_{q+1} z_0 + x_{q+1} y_0 z_0 + \sum_{i=1}^{q} (x_i y_{\sigma(i)} z_0 + x_i y_0 z_i + x_0 y_i z_i).$$

The usual Coppersmith-Winograd tensor CW_q results by setting σ to the identity permutation. Just as in [2, Section 7.1], we can see that Theorems 16 and 18 immediately apply to $CW_{q,\sigma}$ to show that there is a universal constant $\delta > 0$ such that for any q and σ we have $\tilde{S}(CW_{q,\sigma}) \leq (q+2)^{1-\delta}$, and hence a universal constant $c > 2$ such that $\omega_u(CW_{q,\sigma}) \geq c$. Indeed, by proceeding in this way, we get the exact same constants as in [2].

That said, we will now use Theorem 18 to prove that $c \geq 2.16805$. (In fact, essentially the same argument as we present now shows that [2, Theorem 5.2] was already sufficient to show the weaker claim that $\omega_g(CW_{q,\sigma}) \geq 2.16805$).

We begin by partitioning the variable sets of $CW_{q,\sigma}$, using the notation of Theorem 18. Let $X_0 = \{x_0\}$, $X_1 = \{x_1, \ldots, x_q\}$, and $X_2 = \{x_{q+1}\}$, so that $X_0 \cup X_1 \cup X_2$ is a partition of the x-variables of $CW_{q,\sigma}$.[5] Similarly, let $Y_0 = \{y_0\}$, $Y_1 = \{y_1, \ldots, y_q\}$, $Y_2 = \{y_{q+1}\}$, $Z_0 = \{z_0\}$, $Z_1 = \{z_1, \ldots, z_q\}$, and $Z_2 = \{z_{q+1}\}$. We can see this is a $CW_{q,\sigma}$-symmetric partition with $L = \{T_{002}, T_{020}, T_{200}, T_{011}, T_{101}, T_{110}\}$.

Consider any probability distribution $p \in P^{sym}(L)$. By symmetry, we know that $p(T_{002}) = p(T_{020}) = p(T_{200}) = v$ and $p(T_{011}) = p(T_{101}) = p(T_{110}) = 1/3 - v$ for some value $v \in [0, 1/3]$. Applying Theorem 18, and in particular Proposition 21, yields:

$$\tilde{S}(CW_q) \leq \sup_{v \in [0,1/3]} \frac{q^{2(1/3-v)}}{v^v (2/3-2v)^{2/3-2v} (1/3+v)^{1/3+v}}.$$

[5] The sets of partitions were 1-indexed before, but we 0-index here for notational consistency with past work.

In fact, we will see in the next section that this is tight (i.e. the value above is *equal* to $\tilde{S}(CW_q)$, not just an upper bound on it). The values for the first few q can be computed using optimization software as follows:

q	$\tilde{S}(CW_{q,\sigma})$
1	$2.7551\cdots$
2	$3.57165\cdots$
3	$4.34413\cdots$
4	$5.07744\cdots$
5	$5.77629\cdots$
6	$6.44493\cdots$
7	$7.08706\cdots$
8	$7.70581\cdots$

Finally, using the lower bound $\tilde{R}(CW_{q,\sigma}) \geq q+2$ (in fact, it is known that $\tilde{R}(CW_{q,\sigma}) = q+2$), and the upper bound on $\tilde{S}(CW_{q,\sigma})$ we just proved, we can apply Theorem 14 to give lower bounds $\omega_u(CW_{q,\sigma}) \geq 2\log(\tilde{R}(CW_{q,\sigma}))/\log(\tilde{S}(CW_{q,\sigma})) \geq 2\log(q+2)/\log(\tilde{S}(CW_{q,\sigma}))$ as follows:

q	Lower Bound on $\omega_u(CW_{q,\sigma})$
1	$2.16805\cdots$
2	$2.17794\cdots$
3	$2.19146\cdots$
4	$2.20550\cdots$
5	$2.21912\cdots$
6	$2.23200\cdots$
7	$2.24404\cdots$
8	$2.25525\cdots$

It is not hard to see that the resulting lower bound on $\omega_u(CW_{q,\sigma})$ is increasing with q and is always at least $2.16805\ldots$ (see Appendix A below for a proof), and hence that for any q and any σ we have $\omega_u(CW_{q,\sigma}) \geq 2.16805$ as desired.

5.2 Generalized Simple Coppersmith-Winograd Tensors

Similar to $CW_{q,\sigma}$, we can define for a positive integer q and a permutation $\sigma : [q] \to [q]$ the simple Coppersmith-Winograd tensor $cw_{q,\sigma}$ given by:

$$cw_{q,\sigma} := \sum_{i=1}^{q} (x_i y_{\sigma(i)} z_0 + x_i y_0 z_i + x_0 y_i z_i).$$

These tensors, when σ is the identity permutation id, are well-studied. For instance, Coppersmith and Winograd [12] showed that if $\tilde{R}(cw_{2,id}) = 2$ then $\omega = 2$.

We will again give a tight bound on $\tilde{S}(cw_{q,\sigma})$ using Theorem 18 combined with the next section. To apply Theorem 18, and in particular Proposition 21, we again pick a partition of the variables. Let $X_0 = \{x_0\}$, $X_1 = \{x_1, \ldots, x_q\}$, $Y_0 = \{y_0\}$, $Y_1 = \{y_1, \ldots, y_q\}$, $Z_0 = \{z_0\}$, and $Z_1 = \{z_1, \ldots, z_q\}$. This is a $cw_{q,\sigma}$-symmetric partition with $L = \{T_{011}, T_{101}, T_{110}\}$. There is a unique $p \in P^{sym}(L)$, which assigns probability $1/3$ to each part. It follows that

$$\tilde{S}(cw_{q,\sigma}) \leq (1/3)^{-1/3}(2/3)^{-2/3} \cdot q^{2/3} = \frac{3}{2^{2/3}} \cdot q^{2/3}.$$

Again, we will see in the next section that this bound is tight. Using the lower bound $\tilde{R}(cw_{q,\sigma}) \geq q+1$, we get the lower bound

$$\omega_u(cw_{q,\sigma}) \geq 2\frac{\log(q+1)}{\log\left(\frac{3}{2^{2/3}} \cdot q^{2/3}\right)}.$$

The first few values are as follows; note that we cannot get a bound better than 2 when $q = 2$ because of Coppersmith and Winograd's remark.

q	Lower Bound on $\omega_u(cw_{q,\sigma})$
1	$2.17795\cdots$
2	2
3	$2.02538\cdots$
4	$2.06244\cdots$
5	$2.09627\cdots$
6	$2.12549\cdots$
7	$2.15064\cdots$

5.3 Cyclic Group Tensors

We next look at two tensors which were studied in [9], [1], and [2, Section 7.3]. For each positive integer q, define the tensor T_q (the structural tensor of the cyclic group C_q) as:

$$T_q = \sum_{i=0}^{q-1}\sum_{j=0}^{q-1} x_i y_j z_{i+j \bmod q}.$$

Define also the lower triangular version of T_q, called T_q^{lower}, as:

$$T_q^{lower} = \sum_{i=0}^{q-1}\sum_{j=0}^{q-1-i} x_i y_j z_{i+j}.$$

While Theorem 18 does not give any nontrivial upper bounds on $\tilde{S}(T_q)$, it does give nontrivial upper bounds on $\tilde{S}(T_q^{lower})$, as noted in [2, Section 7.3]. Using computer optimization software, we can compute our lower bound on $\tilde{S}(T_q^{lower})$, using Theorem 18 where each partition contains exactly one variable, for the first few values of q:

q	Upper Bound on $\tilde{S}(T_q^{lower})$
2	$1.88988\cdots$
3	$2.75510\cdots$
4	$3.61071\cdots$
5	$4.46157\cdots$

We show in the next section that these numbers are also tight. It is known (see e.g. [1]) that $\tilde{R}(T_q) = \tilde{R}(T_q^{lower}) = q$. Thus we get the following lower bounds on $\omega_u(T_q^{lower}) \geq 2\log(q)/\log(\tilde{S}(T_q^{lower}))$:

q	Lower Bound on $\omega_u(T_q^{lower})$
2	$2.17795\cdots$
3	$2.16805\cdots$
4	$2.15949\cdots$
5	$2.15237\cdots$

These numbers match the lower bounds obtained by [1, 5] in their study of T_q; our Theorem 18 can be viewed as an alternate tool to achieve those lower bounds. The bound approaches 2 as $q \to \infty$, as it is known that $\log(\mathrm{S}(T_q))/\log(q) = 1 - o(1)$ as $q \to \infty$. Interestingly, it is shown in [7, Theorem 4.16] that T_q^{lower} degenerates to T_q over the field $\mathbb{F}_q$, which implies that our bounds above also hold for T_q over $\mathbb{F}_q$.

5.4 The Value of the Subtensor t_{112} of $CW_q^{\otimes 2}$

A key tensor which arises in applying the Laser method to increasing powers of CW_q, including [12, 26, 18, 17, 14], is the tensor t_{112} which (for a given positive integer q) is given by

$$t_{112} := \sum_{i=1}^{q} x_{i,0}y_{i,0}z_{0,q+1} + \sum_{k=1}^{q} x_{0,k}y_{0,k}z_{q+1,0} + \sum_{i,k=1}^{q} x_{i,0}y_{0,k}z_{i,k} + \sum_{i,k=1}^{q} x_{0,k}y_{i,0}z_{i,k}.$$

Coppersmith-Winograd [12] and future work studied the value of this tensor. In [12] it is shown that for every $\tau \in [2/3, 1]$,

$$V_\tau(t_{112}) \geq 2^{2/3}q^\tau(q^{3\tau}+2)^{1/3}.$$

This bound has been used in all the subsequent work using CW_q, without improvement. Here we show it is tight and cannot be improved in the case $\tau = 2/3$:

▶ **Proposition 24.** $V_{2/3}(t_{112}) = 2^{2/3}q^{2/3}(q^2+2)^{1/3}$.

Proof. Consider the variable-symmetric tensor $t_s := t_{112} \otimes rot(t_{112}) \otimes rot(rot(t_{112}))$. As in [12], by definition of $V_{2/3}$, for every $\delta > 0$ there is a positive integer n such that $t_s^{\otimes n}$ has a degeneration to $\bigoplus_i \langle a_i, a_i, a_i\rangle$ for values such that $\sum_i a_i^2 \geq (V_{2/3}(T_{112}))^{3n(1-\delta)}$. In particular, by Corollary 11 this yields the bound

$$\tilde{S}(t_s^{\otimes n}) \geq \sum_i a_i^2 \geq (V_{2/3}(t_{112}))^{3n(1-\delta)}.$$

Since this holds for all $\delta > 0$, it follows that $\tilde{S}(t_s) \geq (V_{2/3}(t_{112}))^3 \geq 2^2q^2(q^2+2)$.

We now upper bound $\tilde{S}(t_s)$ using Theorem 18. Although we are analyzing t_s, we will make use of a partition of the variables of t_{112}. The partition is as follows: $X_0 = \{x_{i,0} \mid i \in [q]\}$, $X_1 = \{x_{0,k} \mid k \in [q]\}$, $Y_0 = \{y_{i,0} \mid i \in [q]\}$, $Y_1 = \{y_{0,k} \mid k \in [q]\}$, $Z_0 = \{z_{i,k} \mid i,k \in [q]\}$, $Z_1 = \{z_{0,q+1}\}$, and $Z_2 = \{z_{q+1,0}\}$. Hence, $L = \{T_{001}, T_{112}, T_{010}, T_{100}\}$. As in [12], and similar to Proposition 21, since t_s is defined as $t_s := t_{112} \otimes rot(t_{112}) \otimes rot(rot(t_{112}))$, it follows that $\tilde{S}(t_s) \leq \limsup_{p \in P(L)} p_X \cdot p_Y \cdot p_Z$. We can assume, again by symmetry, that any probability distribution p on L assigns the same value v to T_{010} and T_{100}, and the same value $1/2 - v$ to T_{001} and T_{112}. We finally get the bound:

$$\tilde{S}(t_s) \leq \limsup_{v \in [0,1/2]} (2q)^2 \cdot \frac{(q^2)^{2v}}{(2v)^{2v}(1/2-v)^{1-2v}}.$$

This is maximized at $v = q^2/(2q^2+2)$, which yields exactly $\tilde{S}(t_s) \leq 2^2q^2(q^2+2)$. The desired bound follows. ◀

The only upper bound we are able to prove on V_τ for $\tau > 2/3$ is the straightforward $V_\tau(t_{112}) \leq V_{2/3}(t_{112})^{3\tau/2} = 2^\tau q^\tau(q^2+2)^{\tau/2}$, which is slightly worse than the best known lower bound $V_\tau(t_{112}) \geq 2^{2/3}q^\tau(q^{3\tau}+2)^{1/3}$. It is an interesting open problem to prove tight

upper bounds on $V_\tau(T)$ for any nontrivial tensor T and value $\tau > 2/3$. $T = t_{112}$ may be a good candidate since the Laser method seems unable to improve $V_\tau(t_{112})$ for any τ, even when applied to any small tensor power $t_{112}^{\otimes n}$.

Notice that we were able to prove a tight bound on $\tilde{S}(t_s)$ here: the upper bound we proved matches a lower bound which we were able to derive from Coppersmith-Winograd's analysis (which made use of the Laser Method) of $V_\tau(t_{112})$. In the next section we will substantially generalize this fact, by showing a tight bound on $\tilde{S}(T)$ for any tensor T to which the Laser Method applies.

6 Slice Rank Lower Bounds via the Laser Method

In this section, we show that the Laser Method can be used to give matching upper and lower bounds on $\tilde{S}(T)$ for any tensor T to which it applies. We will build off of Theorem 18, which we will show matches the bounds which arise in the Laser Method.

Consider any tensor T which is minimal over X, Y, Z, and let $X = X_1 \cup \cdots \cup X_{k_X}$, $Y = Y_1 \cup \cdots \cup Y_{k_Y}$, $Z = Z_1 \cup \cdots \cup Z_{k_Z}$ be partitions of the three variable sets. Define T_{ijk}, L, and p_X for a probability distribution p on L, as in the top of Subsection 3.7. Recall in particular that T_{ijk} is T restricted to the variable sets X_i, Y_j, and Z_k.

▶ **Definition 25.** *We say that T, along with partitions of X, Y, Z, is a* laser-ready tensor partition *if the following three conditions are satisfied:*

(1) *For every $(i, j, k) \in [k_X] \times [k_Y] \times [k_Z]$, either $T_{ijk} = 0$, or else T_{ijk} has a degeneration to a tensor $\langle a, b, c\rangle$ with $ab = |X_i|$, $bc = |Y_j|$, and $ca = |Z_k|$ (i.e. a matrix multiplication tensor which is as big as possible given $|X_i|$, $|Y_j|$, and $|Z_k|$).*

(2) *There is an integer ℓ such that $T_{ijk} \neq 0$ only if $i + j + k = \ell$.*

(3) *T is variable-symmetric, and the partitions are T-symmetric.*

These conditions are exactly those for which the original Laser Method used by Coppersmith and Winograd [12] applies to T. We note that condition (3) is a simplifying assumption rather than a real condition on T: for any tensor T and partitions satisfying conditions (1) and (2), the tensor $T' := T \otimes rot(T) \otimes rot(rot(T))$ along with the corresponding product partitions, satisfies all three conditions, gives at least as good a bound on ω using the Laser Method as T and the original partitions, and more generally has $\omega_u(T') \leq \omega_u(T)$.

▶ **Theorem 26** ([12, 13, 26])**.** *Suppose T, along with the partitions of X, Y, Z, is a laser-ready tensor partition. Then, for any distribution $p \in P^{sym}(L)$, and any positive integer n, the tensor $T^{\otimes n}$ has a degeneration into*

$$\left(\prod_{i \in [k_X]} p(X_i)^{-p(X_i)}\right)^{n - o(n)} \odot \langle a, a, a\rangle,$$

where

$$a = \left(\prod_{T_{ijk} \in L} |X_i|^{p(T_{ijk})}\right)^{n/2 - o(n)}.$$

Proof. Typically, as described in [26, Section 3], there is an additional loss in the size of the degeneration if there are multiple different distributions p, p' with the same marginals (meaning $p(X_i) = p'(X_i)$, $p(Y_j) = p'(Y_j)$, and $p(Z_k) = p'(Z_k)$ for all i, j, k) but different

values of $V(p) := \prod_{T_{ijk} \in L} V_\tau(T_{ijk})^{p(T_{ijk})}$ for any $\tau \in [2/3, 1]$. However, because of condition (1) in the definition of a laser-ready tensor partition, the quantity $V(p)$ is equal to

$$\prod_{T_{ijk} \in L} (|X_i| \cdot |Y_j| \cdot |Z_k|)^{p(T_{ijk}) \cdot \tau/2},$$

and in particular satisfies $V(p) = V(p')$ for any two distributions p, p' with the same marginals. Thus, we do not incur this loss, and we get the desired degeneration. ◀

Our key new result about such tensor partitions is as follows:

▶ **Theorem 27.** *Suppose tensor T, along with the partitions of X, Y, Z, is a laser-ready tensor partition. Then,*

$$\tilde{S}(T) = \limsup_{p \in P^{sym}(L)} p_X.$$

Proof. The upper bound, $\tilde{S}(T) \leq \limsup_{p \in P^{sym}(L)} p_X$, is given by Proposition 21.

For the lower bound, we know from Theorem 26 that for all $p \in P^{sym}(L)$, and all positive integers n, the tensor $T^{\otimes n}$ has a degeneration into

$$\left(\prod_{i \in [k_X]} p(X_i)^{-p(X_i)} \right)^{n - o(n)} \odot \langle a, a, a \rangle,$$

where

$$a = \left(\prod_{T_{ijk} \in L} |X_i|^{p(T_{ijk})} \right)^{n/2 - o(n)}.$$

By Proposition 10, this means $T^{\otimes n}$ has a degeneration to an independent tensor of size

$$\left(\prod_{i \in [k_X]} p(X_i)^{-p(X_i)} \right)^{n - o(n)} \cdot a^2 = p_X^{n - o(n)}.$$

Applying Propositions 8 and 9 implies that $\tilde{S}(T) \geq p_X$ for all $p \in P^{sym}(L)$, as desired. ◀

▶ **Corollary 28.** *The upper bounds on $\tilde{S}(CW_{q,\sigma})$, $\tilde{S}(cw_{q,\sigma})$, $\tilde{S}(T_q^{lower})$, and $\tilde{S}(T_q)$ from Section 5 are tight.*

Proof. $CW_{q,\sigma}$, $cw_{q,\sigma}$, and T_q^{lower}, partitioned as they were in the previous section, are laser-ready tensor partitions. The tight bound for T_q follows from the degeneration to T_q^{lower} described in the previous section. ◀

▶ **Corollary 29.** *Every tensor T with a laser-ready tensor partition (including $CW_{q,\sigma}$, $cw_{q,\sigma}$, and T_q^{lower}) has $\tilde{S}(T) = \tilde{Q}(T)$.*

Proof. All tensors satisfy $\tilde{S}(T) \geq \tilde{Q}(T)$. In Theorem 27, the upper bound on $\tilde{S}(T)$ showed that $T^{\otimes n}$ has a degeneration to an independent tensor of size $\tilde{S}(T)^{n - o(n)}$, which implies that $\tilde{Q}(T) \geq \tilde{S}(T)$. ◀

▶ **Corollary 30.** *If T is a tensor with a laser-ready tensor partition, and applying the Laser method to T with this partition yields an upper bound on ω of $\omega_u(T) \leq c$ for some $c > 2$, then $\omega_u(T) > 2$.*

Proof. When the Laser method shows, as in Theorem 26, that $T^{\otimes n}$ has a degeneration into

$$\left(\prod_{i\in[k_X]} p(X_i)^{-p(X_i)}\right)^{n-o(n)} \odot \langle a,a,a\rangle,$$

the resulting upper bound on $\omega_u(T)$ is that

$$\left(\prod_{i\in[k_X]} p(X_i)^{-p(X_i)}\right)^{n-o(n)} \cdot a^{\omega_u(T)} \geq \tilde{R}(T)^n.$$

In particular, since the left-hand side equals p_X when $\omega_u(T) = 2$, this yields $\omega_u(T) = 2$ if and only if $p_X = \tilde{R}(T)$, so if it yields $\omega_u(T) \leq c$, then $\tilde{S}(T) = p_X < \tilde{R}(T)^{1-\delta}$ for some $\delta > 0$. Combined with Theorem 12 or Theorem 14, this means that $\omega_u(T) > 2$. ◀

References

1. Josh Alman and Virginia Vassilevska Williams. Further limitations of the known approaches for matrix multiplication. In *ITCS*, pages 25:1–25:15, 2018.
2. Josh Alman and Virginia Vassilevska Williams. Limits on all known (and some unknown) approaches to matrix multiplication. In *FOCS*, pages 580–591, 2018.
3. Andris Ambainis, Yuval Filmus, and François Le Gall. Fast matrix multiplication: limitations of the Coppersmith-Winograd method. In *STOC*, pages 585–593, 2015.
4. Dario Bini. Border rank of ap× q× 2 tensor and the optimal approximation of a pair of bilinear forms. In *International Colloquium on Automata, Languages, and Programming*, pages 98–108. Springer, 1980.
5. 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*, 2017(3):1–27, 2017.
6. Matthias Christandl, Péter Vrana, and Jeroen Zuiddam. Barriers for fast matrix multiplication from irreversibility. *arXiv e-prints*, page arXiv:1812.06952, December 2018. `arXiv:1812.06952`.
7. Matthias Christandl, Péter Vrana, and Jeroen Zuiddam. Universal points in the asymptotic spectrum of tensors. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 289–296. ACM, 2018.
8. Michael B Cohen, Yin Tat Lee, and Zhao Song. Solving Linear Programs in the Current Matrix Multiplication Time. *arXiv preprint arXiv:1810.07896; to appear in STOC 2019*, 2018. `arXiv:1810.07896`.
9. Henry Cohn and Christopher Umans. A group-theoretic approach to fast matrix multiplication. In *FOCS*, pages 438–449, 2003.
10. Henry Cohn and Christopher Umans. Fast matrix multiplication using coherent configurations. In *Proceedings of the twenty-fourth annual ACM-SIAM symposium on Discrete algorithms*, pages 1074–1086. Society for Industrial and Applied Mathematics, 2013.
11. Don Coppersmith and Shmuel Winograd. On the Asymptotic Complexity of Matrix Multiplication. *SIAM J. Comput.*, 11(3):472–492, 1982.
12. Don Coppersmith and Shmuel Winograd. Matrix multiplication via arithmetic progressions. *Journal of symbolic computation*, 9(3):251–280, 1990.
13. A.M. Davie and A. J. Stothers. Improved bound for complexity of matrix multiplication. *Proceedings of the Royal Society of Edinburgh, Section: A Mathematics*, 143:351–369, April 2013.
14. Francois Le Gall and Florent Urrutia. Improved Rectangular Matrix Multiplication using Powers of the Coppersmith-Winograd Tensor. In *SODA*, pages 1029–1046, 2018. `doi:10.1137/1.9781611975031.67`.

15 Matti Karppa and Petteri Kaski. Probabilistic Tensors and Opportunistic Boolean Matrix Multiplication. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 496–515. SIAM, 2019.

16 Robert Kleinberg, Will Sawin, and David Speyer. The growth rate of tri-colored sum-free sets. *Discrete Analysis*, June 2018. doi:10.19086/da.3734.

17 François Le Gall. Faster algorithms for rectangular matrix multiplication. In *FOCS*, pages 514–523, 2012.

18 François Le Gall. Powers of tensors and fast matrix multiplication. In *ISSAC*, pages 296–303, 2014.

19 A. Schönhage. Partial and Total Matrix Multiplication. *SIAM J. Comput.*, 10(3):434–455, 1981.

20 V. Strassen. Relative bilinear complexity and matrix multiplication. *J. reine angew. Math. (Crelles Journal)*, 375–376:406–443, 1987.

21 Volker Strassen. Gaussian elimination is not optimal. *Numerische mathematik*, 13(4):354–356, 1969.

22 Volker Strassen. The Asymptotic Spectrum of Tensors and the Exponent of Matrix Multiplication. In *FOCS*, pages 49–54, 1986.

23 Volker Strassen. Degeneration and complexity of bilinear maps: some asymptotic spectra. *Crelles J. Reine Angew. Math*, 413:127–180, 1991.

24 Terence Tao. A symmetric formulation of the Croot-Lev-Pach-Ellenberg-Gijswijt capset bound. https://terrytao.wordpress.com/2016/05/18/a-symmetric-formulation-of-the-croot-lev-pach-ellenberg-gijswijt-capset-bound/, 2016.

25 Terence Tao and Will Sawin. Notes on the "slice rank" of tensors. https://terrytao.wordpress.com/2016/08/24/notes-on-the-slice-rank-of-tensors/, 2016.

26 Virginia Vassilevska Williams. Multiplying matrices faster than Coppersmith-Winograd. In *STOC*, pages 887–898, 2012.

A Proof that $\omega_u(CW_{q,\sigma}) \geq 2.16805$ for all q

Define the function $f : [0, 1/3] \to \mathbb{R}$ by

$$f(v) := \frac{1}{v^v(2/3 - 2v)^{2/3-2v}(1/3 + v)^{1/3+v}}.$$

In Section 5.1, we showed that

$$\omega_u(CW_{q,\sigma}) \geq \min_{v\in[0,1/3]} 2\frac{\log(q+2)}{\log(q^{2/3-2v} \cdot f(v))}.$$

The value of this optimization problem is computed for $1 \leq q \leq 8$ in a table in Section 5.1, where we see that $\omega_u(CW_{q,\sigma}) \geq 2.16805$ for all $q \leq 8$.

Let v_q denote the argmin for the optimization problem. In particular, for $q = 8$, the argmin is $v_8 = 0.017732422\ldots$. From the $q^{2/3-2v}$ term in the optimization problem, we see that $v_{q+1} \leq v_q$ for all q, and in particular, $v_q \leq v_8$ for all $q > 8$. It follows that $f(v_q) \leq f(v_8) = 2.07389\ldots$ for all $q > 8$. Thus, for all $q > 8$ we have:

$$\omega_u(CW_{q,\sigma}) \geq \min_{v\in[0,1/3]} 2\frac{\log(q+2)}{\log(q^{2/3-2v} \cdot f(v_8))} = 2\frac{\log(q+2)}{\log(q^{2/3} \cdot f(v_8))}.$$

This expression equals $2.18562\ldots$ at $q = 9$, and is easily seen to be increasing with q for $q > 9$, which implies as desired that $\omega_u(CW_{q,\sigma}) \geq 2.16805$ for all $q \geq 9$ and hence all q.

Optimality of Linear Sketching Under Modular Updates

Kaave Hosseini
University of California, San Diego, USA
http://cseweb.ucsd.edu/~skhossei/
skhossei@ucsd.edu

Shachar Lovett
University of California, San Diego, USA
https://cseweb.ucsd.edu/~slovett/home.html
slovett@ucsd.edu

Grigory Yaroslavtsev
Indiana University, Bloomington, USA
https://cseweb.ucsd.edu/~slovett/home.html
grigory.yaroslavtsev@gmail.com

Abstract

We study the relation between streaming algorithms and linear sketching algorithms, in the context of binary updates. We show that for inputs in n dimensions, the existence of efficient streaming algorithms which can process $\Omega(n^2)$ updates implies efficient linear sketching algorithms with comparable cost. This improves upon the previous work of Li, Nguyen and Woodruff [23] and Ai, Hu, Li and Woodruff [3] which required a triple-exponential number of updates to achieve a similar result for updates over integers. We extend our results to updates modulo p for integers $p \geq 2$, and to approximation instead of exact computation.

2012 ACM Subject Classification Theory of computation → Communication complexity; Theory of computation → Sketching and sampling

Keywords and phrases communication complexity, linear sketching, streaming algorithm

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.13

Funding *Kaave Hosseini*: Supported by NSF grant CCF-1614023.
Shachar Lovett: Supported by NSF grant CCF-1614023.

1 Introduction

Linear sketching has emerged in the recent years as a fundamental primitive for algorithm design and analysis including streaming and distributed computing. Applications of linear sketching include randomized algorithms for numerical linear algebra (see survey [32]), graph sparsification (see survey [24]), frequency estimation [4], dimensionality reduction [19], various forms of sampling, signal processing, and communication complexity. In fact, linear sketching has been shown to achieve optimal space complexity [3, 23] for processing very long dynamic data streams, which allow elements to be both inserted and deleted. Linear sketching is also a frequently used tool in distributed computing – summaries communicated between processors in massively parallel computational settings are often linear sketches.

In this paper we focus on linear sketches for functions evaluated modulo p. Namely, functions of the form $f\colon \mathbb{Z}_p^n \to [0,1]$. Informally, the main result of our work is that for computing such functions linear sketching modulo p achieves almost optimal space complexity in dynamic streaming and distributed simultaneous communication settings. In particular, the setting of p a power of two (say, 32 or 64) is relevant as CPUs perform computations modulo such powers of two.

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka; Article No. 13; pp. 13:1–13:17


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

Exact sketching for binary data

We start with presenting our result in the simplest setting, where $p = 2$ and where the output of f is binary. Namely, we are interested in computing a given Boolean function of the form $f(x)\colon \{0,1\}^n \to \{0,1\}$ using only a small sketch of the input. In this context it is natural to consider sketches which are linear functions over the finite field $\mathbb{F}_2$. Due to their prominence in design of dynamic streaming graph algorithms and other applications [1, 2, 5, 6, 9, 11, 12, 14, 15, 21, 22, 25] a study of such $\mathbb{F}_2$-sketches has been initiated in [20].

▶ **Definition 1** (Exact $\mathbb{F}_2$-sketching, [20]). *The* exact randomized $\mathbb{F}_2$-sketch complexity *with error δ of a function $f\colon \mathbb{F}_2^n \to \{0,1\}$ is the smallest integer k such that there exists a distribution over linear functions $\ell_1, \ldots, \ell_k : \mathbb{F}_2^n \to \mathbb{F}_2$ and a post-processing function $h : \mathbb{F}_2^k \to \{0,1\}$ that satisfies:*

$$\forall x \in \mathbb{F}_2^n\colon \Pr_{\ell_1,\ldots,\ell_k,h}[h(\ell_1(x), \ell_2(x), \ldots, \ell_k(x)) = f(x)] \geq 1 - \delta.$$

In particular, $\mathbb{F}_2$-sketches naturally allow one to design algorithms for processing data streams in the XOR update model [30] which we refer to as just XOR streams below. In this model the input $x \in \{0,1\}^n$ is generated via a sequence of additive updates to its coordinates $i_1, \ldots, i_t$ where each $i_j \in [n]$. Formally, let $x_0 = 0^n$ and let $x_j = x_{j-1} \oplus e_{i_j}$ where e_k is the k-th unit vector. This corresponds to flipping the bit in position i_j in x at time j and after applying the sequence of updates the resulting input is $x = x_t$. The goal of the streaming algorithm is to output $f(x)$. It is easy to see that by flipping linear functions which depend on x_{i_j} when the update i_j arrives one can maintain an $\mathbb{F}_2$-sketch through the XOR stream. Hence the size of the $\mathbb{F}_2$-sketch gives an upper bound on the space complexity of streaming algorithms in XOR streams.[1]

Whether this simple approach in fact achieves optimal space complexity for streaming applications is one of the central questions in the field. Two structural results regarding space optimality $\mathbb{F}_2$-sketching for dynamic streaming are known:

1. $\mathbb{F}_2$-sketches achieve optimal space for streams of length $2^{2^{2^{\Omega(n)}}}$ [3, 20, 23].
2. $\mathbb{F}_2$-sketches achieve optimal space for streams of length $\tilde{O}(n)$ under the assumption that updates are uniformly random [20].

It is open whether optimality of $\mathbb{F}_2$-sketching holds for short streams without any assumptions about the distribution of updates. In fact, it was conjectured in [20] that such optimality might hold for streams of length only $2n$ (see also Open Problem 78 on `http://sublinear.info` from Banff Workshop on Communication Complexity and Applications, 2017).

In this paper we make major progress towards resolving the gap between the two results discussed above. In particular, we show the following theorem.

▶ **Theorem 2.** *Let $f\colon \{0,1\}^n \to \{0,1\}$. Assume that there exists a streaming algorithm for computing f over XOR streams of length $\Omega(n^2)$, which uses c bits of space. Then the exact randomized $\mathbb{F}_2$-sketch complexity of f is $O(c)$.*

Moreover, using some more advanced tools in additive combinatorics we prove the following.

[1] More precisely, one also needs to derandomize the randomness in the sketching algorithm. This follows from a standard application of Nisan's pseudorandom generator [27]. For completeness we explain this in Appendix A.

▶ **Theorem 3.** *Let $f\colon \{0,1\}^n \to \{0,1\}$. Assume that there exists a streaming algorithm for computing f over XOR streams of length $\Omega(n)$, which uses c bits of space. Then the exact randomized $\mathbb{F}_2$-sketch complexity of f is $O(c^4)$.*

The proof of Theorems 2 and 3 follows from a standard approach in this field, of proving lower bounds for one-way communication protocols. We refer the reader to Section 2 where the model is defined, and to Theorems 8 and 12 which are the formal versions of Theorems 2 and 3, respectively.

Extensions to updates modulo p

We now consider the more general streaming model where updates modulo p are allowed, where $p \geq 2$ is an integer. In this model an underlying n-dimensional vector x is initialized to 0^n and evolves through a sequence of additive updates to its coordinates. These updates are presented to the streaming algorithm as a sequence and have the form $x_i \leftarrow (x_i + \delta_t) \mod p$ changing the i-th coordinate by an additive increment δ_t modulo p in the t-th update. Here δ_t can be an arbitrary positive or negative integer. In this setting the streaming algorithm is required to output a given function f of $\{0,\ldots,p-1\}^n$ in the end of the stream.

The definition of the exact randomized $\mathbb{Z}_p$-sketch complexity of f is the natural extension of the definition for $\mathbb{F}_2$.

▶ **Definition 4** (Exact $\mathbb{Z}_p$-sketching)**.** *The* exact randomized $\mathbb{Z}_p$-sketch complexity *with error δ of a function $f\colon \mathbb{Z}_p^n \to \{0,1\}$ is the smallest integer k such that there exists a distribution over linear functions $\ell_1,\ldots,\ell_k : \mathbb{Z}_p^n \to \mathbb{Z}_p$ and a post-processing function $h : \mathbb{Z}_p^k \to \{0,1\}$ that satisfies:*

$$\forall x \in \mathbb{Z}_p^n\colon \Pr_{\ell_1,\ldots,\ell_k,h}[h(\ell_1(x),\ell_2(x),\ldots,\ell_k(x)) = f(x)] \geq 1-\delta.$$

▶ **Theorem 5.** *Let $f\colon \mathbb{Z}_p^n \to \{0,1\}$. Assume that there exists a streaming algorithm for computing f over streams with modulo p updates of length $\Omega(n^2 \log p)$, which uses c bits of space. Then the exact randomized $\mathbb{Z}_p$-sketch complexity of f is $O(c)$.*

The proof of Theorem 5 for prime p is very similar to the proof of Theorem 2. However, for non-prime p the proof is a bit more involved. We define the relevant model in Section 4, and the relevant theorem that implies Theorem 5 is Theorem 17.

Extensions to approximation

It is also natural to consider real-valued functions $f\colon \{0,1\}^n \to [0,1]$ and to allow the streaming algorithm to compute f with error ϵ (see e.g. [34]). It turns out that technically, a convenient notion of approximation is ℓ_2 approximation. Namely, a randomized function $\mathbf{g}$ (computed by a streaming protocol, or a sketching protocol) ε-approximates f if

$$\mathbb{E}\left[|f(x) - \mathbf{g}(x)|^2\right] \leq \varepsilon \qquad \forall x \in \{0,1\}^n.$$

A similar definition holds for more general functions $f : \mathbb{Z}_p^n \to [0,1]$. The definition of approximate randomized $\mathbb{Z}_p$-sketch complexity is the natural extension of the previous definitions.

▶ **Definition 6** (Approximate $\mathbb{Z}_p$-sketching)**.** *The* approximate randomized $\mathbb{Z}_p$-sketch complexity *with error δ of a function $f\colon \mathbb{Z}_p^n \to [0,1]$ is the smallest integer k such that there exists a distribution over linear functions $\ell_1,\ldots,\ell_k : \mathbb{Z}_p^n \to \mathbb{Z}_p$ and a post-processing function $h : \mathbb{Z}_p^k \to [0,1]$ that satisfies:*

$$\forall x \in \mathbb{Z}_p^n\colon \mathop{\mathbb{E}}_{\ell_1,\ldots,\ell_k,h}\left[|h(\ell_1(x),\ell_2(x),\ldots,\ell_k(x)) - f(x)|^2\right] \leq \delta.$$

▶ **Theorem 7.** *Let $f\colon \mathbb{Z}_p^n \to [0,1]$. Assume that there exists a streaming algorithm which ε-approximates f over streams with modulo p updates of length $\Omega(n^2 \log p)$, which uses c bits of space. Then the approximate randomized $\mathbb{Z}_p$-sketch complexity of f with error $O(\varepsilon)$ is $O(c)$.*

We develop the machinery needed to handle approximation in two steps. First, in Section 3 we prove it for $p = 2$, see in particular Theorem 14. For general p it is done in Section 4, where the relevant theorem which implies Theorem 7 is Theorem 18.

Techniques

The result in Theorem 2 starts with a standard connection between streaming algorithms and a multi-party one-way communication game. Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function, and assume that we are given a streaming algorithm for f which can process streams of Nn updates. We model this as a game with N players, each holding an input in $\{0,1\}^n$ which captures the commulative effect of n binary updates. We denote these inputs as $x_1, \ldots, x_N$.

The players communicate sequentially in N rounds where in the i-th round the i-th player sends a message to the $(i+1)$-th player. The players have access to shared randomness and the i-th message can depend on the $(i-1)$-th message, the input x_i and the shared randomness. The message sent in the final N-th round should be equal to $f(x_1 + \cdots + x_N)$. In fact, our proof works in a more general model where the i-th message can depend on all messages sent by previous players. We refer to the above model as the *one-way broadcasting communication model*.

We show (Theorem 8) that in any protocol which computes f at least one of the messages sent by the players has to be of size $\Omega(k)$ where k is the smallest dimension of an $\mathbb{F}_2$-sketch for f. This immediately implies a space lower bound of $\Omega(k)$ for streaming algorithms in the XOR update model. Indeed, if a streaming algorithm with smaller space existed then the players could just pass its state as their message after applying updates corresponding to their local inputs.

Our proof of the communication lower bound proceeds as follows. First, it will be easier to present the argument for $N+1$ players instead of N players. Assume that there exists a communication protocol which succeeds with probability q and sends at most c bits in every round. This protocol has to succeed for any distribution of the inputs. So, fix a "hard" distribution D over inputs $x \in \{0,1\}^n$.

We sample the inputs $x_1, \ldots, x_{N+1} \in \{0,1\}^n$ to the N players as follows: first, sample $x \sim D$. Then, sample $x_1, \ldots, x_N \in \{0,1\}^n$ uniformly. Finally, set $x_{N+1} = x + x_1 + \ldots + x_N$, so that the sum (modulo two) of the inputs to the $N+1$ players equals x.

Next, an averaging argument then shows that there is a transcript (sequence of messages) $\pi = (m_1, \ldots, m_N)$ of the first N players such that:

- **(i)** Conditioned on the transcript π, the protocol computes f correctly with probability $\approx q$.
- **(ii)** The probability for the transcript π is not too tiny, concretely $\approx 2^{-cN}$.

Once we fixed the transcript π, note that the output of the protocol depends only on the last input x_{N+1}, which we denote by $F(x_{N+1})$. Recall that by our construction, $x_{N+1} = x + x_1 + \ldots + x_N$. Define sets $A_1, \ldots, A_N \subset \{0,1\}^n$ such that whenever $x_1 \in A_1, \ldots, x_N \in A_N$, the players send the transcript π. By (ii) above it holds that the density of a typical A_i is approximately 2^{-c}. Then, if we sample $y_i \in A_i$ uniformly then by (i) we have

$$\Pr_{x \sim D, y_i \in A_i} [F(x + y_1 + \ldots + y_N) = f(x)] \approx q.$$

The next step is to apply Fourier analysis. In particular, we rely of Chang's lemma [10]. This allows us to deduce that there exists a subspace $V \subset \mathbb{F}_2^n$ of co-dimension $O(c)$ such that, if we sample in addition $v \in V$ uniformly, then

$$\Pr_{x\sim D, y_i \in A_i, v\in V}[F(x + y_1 + \ldots + y_N + v) = f(x)] \approx q.$$

Concretely, V is chosen to be orthogonal to the common large Fourier coefficients of the indicator functions of $A_1, \ldots, A_N$. In order for this to hold, it is necessary to choose N large enough so that the sum $y_1 + \ldots + y_N$ "mixes" enough in the group $\mathbb{F}_2^n$. It turns out that $N = \Omega(n)$ is sufficient for this.

This allows us to define a randomized $\mathbb{F}_2$-sketching protocol. Consider the quantity

$$g(x) = \Pr_{y_i \in A_i, v\in V}[F(x + y_1 + \ldots + y_N + v) = 1].$$

The function $g(x)$ depends only on the coset $x + V$, and hence can be computed by a randomized $\mathbb{F}_2$-sketching protocol with complexity equals to the co-dimension of V.

The results for updates modulo p (Theorem 5) and approximation (Theorem 7) follow the same general scheme, except that now players are holding inputs in $\mathbb{Z}_p^n$ and we convert any c-bit protocol with small error into a sketch modulo p of dimension $O(c)$ which has a similar error. In order to achieve mixing in this setting the required number of players is $N = \Omega(n \log p)$.

Distributed computing in the simultaneous communication model

Our results imply that lower bounds on linear sketches modulo p immediately lead to lower bounds for computing additive functions in the simultaneous communication complexity (SMP) model. In this model [7,8] there are N players and a coordinator, who are all aware of a function $f\colon \mathbb{Z}_p^n \to [0, 1]$. The players have inputs $x_1, \ldots, x_N \in \mathbb{Z}_p^n$ and must send messages of minimal size to the coordinator so that the coordinator can compute $f(x_1, \ldots, x_N)$ using shared randomness. If f is additive, i.e. of the form $f(x_1 + \cdots + x_N)$ then this is strictly harder than the one-way broadcasting model described above. Note that dimension of the best linear sketch modulo p for f still translates to a protocol for the SMP model.

Previous work

Most closely related to ours are results of [23] and [3] which stemmed from the work of [16]. In particular [23] shows that under various assumptions about the updates turnstile streaming algorithms can be turned into linear sketches over integers with only a $O(\log p)$ multiplicative loss in space. While this is similar to our results, these approaches inherently require extremely long streams of adversarial updates (of length triply exponential in n in [23]) as they essentially aim to fail any small space algorithm (modeled as a finite state automaton) using a certain sequence of updates. Furthermore, the results of [23] rely on a certain "box constraint" requirement. This requirement says that correctness of the streaming algorithm should be guaranteed for the resulting input $x \in \{-m, \ldots, m\}^n$ even if the intermediate values of the coordinates of x throughout the stream are allowed to be much larger than m in absolute value. While this requirement has been subsequently removed in [3], their results again impose a certain constraint on the class of streaming algorithms they are applicable to. In particular, their Theorem 3.4 which removes the "box constraint" is only applicable to algorithms which use space at most $\frac{c \log m}{n}$ which is only non-trivial for $m = \Omega(2^n)$.

It has been open since the work of [23] and [3] whether similar results can be obtained using the tools from communication complexity, as has been the case for most other streaming lower bounds. While our results do not apply directly to updates over integers, a key component of the [3, 23] technique is to first reduce general automata to linear sketching modulo fixed integers. Hence our result can be seen as an alternative to their reduction which is specific to modular updates and is obtained through communication complexity tools.

Another related line of work is on communication protocols for XOR-functions [17, 20, 26, 29, 31, 33, 34]. For inputs $x_1, \ldots, x_N \in \mathbb{F}_2^n$ a multi-parity XOR-function is defined as $f(x_1 + \cdots + x_N)$. For the case of $p = 2$ our results are using one-way broadcasting communication complexity for the corresponding XOR-function of interest. While the communication complexity of XOR-functions has been studied extensively in the two-party communication model, to the best of our knowledge prior to our work it has not been considered in the one-way multi-party setting.

2 Sketching for $f : \mathbb{F}_2^n \to \{0, 1\}$

2.1 Model

We use regular letters x, f for deterministic objects and bold letters $\mathbf{x}, \mathbf{f}$ for random variables.

Streaming protocol

Let $F : (\mathbb{F}_2^n)^N \to \{0, 1\}$ be an N-player function, where the players' inputs are $x_1, \ldots, x_N \in \mathbb{F}_2^n$. We assume that the players have access to shared randomness $\mathbf{r} \in \{0, 1\}^r$.

A *streaming protocol* for F with c bits of communication is defined as follows. The players send messages in order, where the i-th player's message $\mathbf{m}_i$ depends on her input x_i, the previous player's message $\mathbf{m}_{i-1}$ and the shared randomness $\mathbf{r}$. That is,

$$\mathbf{m}_1 = M_1(x_1, \mathbf{r}),$$
$$\mathbf{m}_i = M_i(x_i, \mathbf{m}_{i-1}, \mathbf{r}), \qquad i = 2, \ldots, N$$

where $M_i : \mathbb{F}_2^n \times \{0, 1\}^c \times \{0, 1\}^r \to \{0, 1\}^c$ for $1 \leq i \leq N-1$ and $M_N : \mathbb{F}_2^n \times \{0, 1\}^c \times \{0, 1\}^r \to \{0, 1\}$, where the output of the protocol is the last message sent $\mathbf{m}_N \in \{0, 1\}$. We may write it as $\mathbf{m}_N = G(x_1, \ldots, x_N, \mathbf{r})$, where G respects the protocol structure:

$$G(x_1, \ldots, x_N, \mathbf{r}) = M_n(\ldots, M_2(x_2, M_1(x_1, \mathbf{r}), \mathbf{r}), \mathbf{r}).$$

The protocol computes F correctly with probability q, if for all possible inputs $x_1, \ldots, x_N \in \mathbb{F}_2^n$, it holds that

$$\Pr\left[G(x_1, \ldots, x_N, \mathbf{r}) = F(x_1, \ldots, x_N)\right] \geq q.$$

One-way broadcasting

A one-way broadcasting protocol is a generalization of a streaming protocol. We introduce this model as our simulation theorem extends to this model seamlessly.

In this model, the message sent by the i-th player is seen by all the players coming after her. Equivalently, the i-th player's message may depend on $\mathbf{m}_1, \ldots, \mathbf{m}_{i-1}$,

$$\mathbf{m}_i = M_i(x_i, \mathbf{m}_1, \ldots, \mathbf{m}_{i-1}, \mathbf{r}).$$

The notion of a protocol computing F correctly with probability q is defined analogously.

Linear sketches

A function $g : \mathbb{F}_2^n \to \{0,1\}$ is a k-linear-junta if $g(x) = h(\ell_1(x), \ldots, \ell_k(x))$, where each $\ell_i : \mathbb{F}_2^n \to \mathbb{F}_2$ is a linear function, and $h : \mathbb{F}_2^k \to \{0,1\}$ is an arbitrary function. A linear sketch of cost k is a distribution $\mathbf{g}$ over k-linear-juntas. We think of $\mathbf{g}$ as a randomized function $\mathbf{g} : \mathbb{F}_2^n \to \{0,1\}$. It computes f with success probability q if, for every input $x \in \mathbb{F}_2^n$, it holds that

$$\Pr[\mathbf{g}(x) = f(x)] \geq q.$$

Simulation theorem

Let $f : \mathbb{F}_2^n \to \{0,1\}$ and $F : (\mathbb{F}_2^n)^N \to \{0,1\}$ be the N-player function defined by

$$F(x_1, \ldots, x_N) = f(x_1 + \ldots + x_N).$$

We show that if there are sufficiently many players (N is large enough) and F has an efficient one-way broadcasting protocol, then f also has an efficient linear sketch.

▶ **Theorem 8.** *Let $N \geq 10n$ and suppose that F has a one-way broadcasting protocol with c bits of communication per message and success probability q. Then there exists a linear sketch of cost k which computes f with success probability $q - 2^{-\Omega(N)}$, where $k = O(c)$.*

▶ Remark 9. We remark that the statement and proof of Theorem 8 generalizes straightforwardly to the case that $f : \mathbb{F}_p^n \to \{0,1\}$ for any prime p; the only difference is that we have to ensure that $N \geq 10n \log p$. We provide the proof over $\mathbb{F}_2$ to slightly simplify the notation.

2.2 Proof of Theorem 8

By Yao's minimax principle, it will suffice to show that for any distribution D over $\mathbb{F}_2^n$, there exists a k-linear-junta $g : \mathbb{F}_2^n \to \{0,1\}$ such that

$$\Pr_{\mathbf{x} \sim D}[g(\mathbf{x}) = f(\mathbf{x})] \geq q - 2^{-\Omega(N)}.$$

Fix a distribution D. We consider the following distribution over the inputs. It will be easier to assume we have $N+1$ players instead of N players. First, sample $\mathbf{x}_1, \ldots, \mathbf{x}_N \in \mathbb{F}_2^n$ uniformly, and let $\mathbf{x} \sim D$. Set $\mathbf{x}_{N+1} = \mathbf{x}_1 + \ldots + \mathbf{x}_N + \mathbf{x}$. Under this input distribution, there exists a fixed choice of the shared randomness which attains success probability $\geq q$. Namely, there is a fixed r^* such that for $\mathbf{m}_1 = M_1(\mathbf{x}_1, r^*), \mathbf{m}_2 = M_2(\mathbf{x}_2, \mathbf{m}_1, r^*)$, etc, it holds that

$$\Pr_{\mathbf{x}_1, \ldots, \mathbf{x}_{N+1}}[G(\mathbf{m}_1, \ldots, \mathbf{m}_{N+1}, r^*) = f(\mathbf{x})] \geq q. \tag{1}$$

Let $\boldsymbol{\pi} = (\mathbf{m}_1, \ldots, \mathbf{m}_N) \in \{0,1\}^{cN}$ denote the messages of the first N players. For every possible value π of $\boldsymbol{\pi}$, define

$$a(\pi) = \Pr[\boldsymbol{\pi} = \pi], \qquad b(\pi) = \Pr[G(\mathbf{m}_1, \ldots, \mathbf{m}_{N+1}, r^*) = f(\mathbf{x}) \mid \boldsymbol{\pi} = \pi].$$

Then we may rewrite Equation (1) as

$$\sum_{\pi} a(\pi) b(\pi) \geq q.$$

Let $\delta = 2^{-N}$. By averaging, there exists a choice of $\pi = (m_1, \ldots, m_N)$ such that

(i) $a(\pi) \geq \delta 2^{-cN} = 2^{-(c+1)N}$.
(ii) $b(\pi) \geq q - \delta$.

Define sets $A_i = \{x_i \in \mathbb{F}_2^n : M_i(x_i, m_1, \ldots, m_{i-1}, r^*) = m_i\}$ for $i = 1, \ldots, N$ so that

$$[\boldsymbol{\pi} = \pi] \quad \Leftrightarrow \quad [\mathbf{x}_1 \in A_1, \ldots, \mathbf{x}_N \in A_N]\,.$$

Let $\alpha_i = \frac{|A_i|}{2^n}$ denote the density of A_i. Condition (i) translates to

$$\prod_{i=1}^{N} \alpha_i = a(\pi) \geq 2^{-(c+1)N}. \tag{2}$$

Next, conditioned on $\boldsymbol{\pi} = \pi$, the one-way broadcasting protocol simplifies. First, define $h : \mathbb{F}_2^n \to \{0,1\}$ as

$$h(x_{N+1}) = G(m_1, \ldots, m_N, M_{N+1}(x_{N+1}, m_1, \ldots, m_N, r^*), r^*).$$

Let $\mathbf{y}_1 \in A_1, \ldots, \mathbf{y}_N \in A_N$ be uniformly and independently chosen. Condition (ii) translates to

$$\Pr\left[h(\mathbf{x} + \mathbf{y}_1 + \ldots + \mathbf{y}_N) = f(\mathbf{x})\right] = b(\pi) \geq q - \delta. \tag{3}$$

We next apply Fourier analysis. Let us quickly set some common notation in the following paragraph.

Fourier analysis

Let $f : \mathbb{F}_2^n \to \mathbb{R}$ be a function. Then given $\gamma \in \mathbb{F}_2^n$, the Fourier coefficient $\widehat{f}(\gamma)$ is define as $\widehat{f}(\gamma) = \mathbb{E}_{x \in \mathbb{F}_2^n} f(x)(-1)^{\langle x, \gamma \rangle}$. The function f can be expressed in the Fourier basis as $f(x) = \sum_{\gamma \in \mathbb{F}_2^n} \widehat{f}(\gamma)(-1)^{\langle x, \gamma \rangle}$. Given two functions $f, g : \mathbb{F}_2^n \to \mathbb{R}$, their convolution $f * g : \mathbb{F}_2^n \to \mathbb{R}$, is defined by $f * g(x) = \mathbb{E}_{y \in \mathbb{F}_2^n} f(y) g(x + y)$. Moreover, we have the equality $\widehat{f * g}(\gamma) = \widehat{f}(\gamma)\widehat{g}(\gamma)$ for all $\gamma \in \mathbb{F}_2^n$. Given a set $A \subset \mathbb{F}_2^n$ of density $\alpha = \frac{|A|}{2^n}$, define its normalized indicator function $\varphi_A : \mathbb{F}_2^n \to \mathbb{R}$ as

$$\varphi_A(x) = \begin{cases} 1/\alpha & \text{if } x \in A \\ 0 & \text{otherwise} \end{cases}.$$

Note that under this normalization, $\widehat{\varphi_A}(0) = \mathbb{E}[\varphi_A] = 1$ and $|\widehat{\varphi_A}(\gamma)| \leq 1$ for all $\gamma \in \mathbb{F}_2^n$.

Going back to the proof, for technical reasons we switch to $\{-1, 1\}$ instead of $\{0, 1\}$. Define the functions $h' = (-1)^h$ and $f' = (-1)^f$. We work with h', f' instead. Note that Equation (3) translates to

$$\Pr\left[h'(\mathbf{x} + \mathbf{y}_1 + \ldots + \mathbf{y}_N) f'(\mathbf{x}) = 1\right] = b(\pi) \geq q - \delta. \tag{4}$$

We use the following immediate consequence of Chang's lemma (lemma 3.12 in [10]).

▶ **Lemma 10.** *Let $A \subset \mathbb{F}_2^n$ of density $\alpha = \frac{|A|}{2^n}$. Let $\gamma_1, \ldots, \gamma_k \in \mathbb{F}_2^n$ be linearly independent. Then*

$$\sum_{i=1}^{k} |\widehat{\varphi_A}(\gamma_i)|^2 \leq 8 \log 1/\alpha.$$

Let $S \subset \mathbb{F}_2^n$ be a set of "noticeable" Fourier coefficients of $A_1, \ldots, A_N$, defined as follows. First, define

$$B = \left\{ i \in [N] : \frac{|A_i|}{2^n} \geq 2^{-2(c+1)} \right\}.$$

Equation (2) implies that $|B| \geq N/2$. Then, define

$$S = \left\{ \gamma \in \mathbb{F}_2^n : \sum_{i \in B} |\widehat{\varphi_{A_i}}(\gamma)|^2 \geq |B|/2 \right\}.$$

Let $\gamma_1, \ldots, \gamma_k$ be a maximal set of linearly independent elements in S. Lemma 10 implies that $k = O(c)$. Thus, there exists a subspace $U \subset \mathbb{F}_2^n$ of dimension k such that $S \subset U$. Let $V \subset \mathbb{F}_2^n$ be the orthogonal subspace to U.

▷ Claim 11. Let $\mathbf{y}_1 \in A_1, \ldots, \mathbf{y}_N \in A_N, \mathbf{v} \in V$ be chosen uniformly and independently. Then for every $x \in \mathbb{F}_2^n$ it holds that

$$|\mathbb{E}[h'(x + \mathbf{y}_1 + \ldots + \mathbf{y}_N)] - \mathbb{E}[h'(x + \mathbf{y}_1 + \ldots + \mathbf{y}_N + \mathbf{v})]| \leq 2^n 2^{-N/8}.$$

Proof. We rewrite both expressions using their Fourier expansion:

$$\mathbb{E}[h'(x + \mathbf{y}_1 + \ldots \mathbf{y}_N)] = \varphi_{A_1} * \cdots * \varphi_{A_N} * h'(x) = \sum_{\gamma \in \mathbb{F}_2^n} (-1)^{\langle \gamma, x \rangle} \widehat{h'}(\gamma) \prod_{i=1}^{N} \widehat{\varphi_{A_i}}(\gamma)$$

and similarly

$$\mathbb{E}[h'(x + \mathbf{y}_1 + \ldots \mathbf{y}_N + \mathbf{v})] = \sum_{\gamma \in \mathbb{F}_2^n} (-1)^{\langle \gamma, x \rangle} \widehat{h'}(\gamma) \widehat{\varphi_V}(\gamma) \prod_{i=1}^{N} \widehat{\varphi_{A_i}}(\gamma).$$

As V is a subspace, we have $\widehat{\varphi_V}(\gamma) = 1_U(\gamma)$. We can thus bound

$$|\mathbb{E}[h'(x + \mathbf{y}_1 + \ldots + \mathbf{y}_N)] - \mathbb{E}[h'(x + \mathbf{y}_1 + \ldots + \mathbf{y}_N + \mathbf{v})]| \leq \sum_{\gamma \notin U} \prod_{i=1}^{N} |\widehat{\varphi_{A_i}}(\gamma)|.$$

If $\gamma \in S$ then also $\gamma \in U$. Otherwise, by the construction of S, $|\widehat{\varphi_{A_i}}(\gamma)|^2 \leq 1/2$ for at least $|B|/2$ elements $i \in [N]$. We thus have

$$|\mathbb{E}[h'(x + \mathbf{y}_1 + \ldots + \mathbf{y}_N)] - \mathbb{E}[h'(x + \mathbf{y}_1 + \ldots + \mathbf{y}_N + \mathbf{v})]| \leq 2^n 2^{-|B|/4} \leq 2^n 2^{-N/8}.$$ ◀

Moreover, using the assumption $N \geq 10n$ provides

$$|\mathbb{E}[h'(x + \mathbf{y}_1 + \ldots + \mathbf{y}_N)] - \mathbb{E}[h'(x + \mathbf{y}_1 + \ldots + \mathbf{y}_N + \mathbf{v})]| \leq 2^{-\Omega(N)}.$$

Now, by Equation (4) we already have

$$\mathbb{E}[h'(\mathbf{x} + \mathbf{y}_1 + \ldots + \mathbf{y}_N) f'(\mathbf{x})] \geq 2q - 1 - 2^{-\Omega(N)}$$

allowing us to obtain

$$\mathbb{E}[h'(\mathbf{x} + \mathbf{y}_1 + \ldots + \mathbf{y}_N + \mathbf{v}) f'(\mathbf{x})] \geq 2q - 1 - 2^{-\Omega(N)}$$

implying

$$\Pr\left[h(\mathbf{x} + \mathbf{y}_1 + \ldots + \mathbf{y}_N + \mathbf{v}) = f(\mathbf{x})\right] \geq q - 2^{-\Omega(N)}$$

To conclude the proof, define the function $w : \mathbb{F}_2^n \to [0,1]$ as

$$w(x) = \mathbb{E}\left[h(x + \mathbf{y}_1 + \ldots + \mathbf{y}_N + \mathbf{v})\right].$$

Note that $w(x) = W(\ell_1(x), \ldots, \ell_k(x))$, where $\ell_1, \ldots, \ell_k$ are a basis for U, and $W : \mathbb{F}_2^k \to [0,1]$. Define a randomized function $\mathbf{G} : \mathbb{F}_2^k \to \{0,1\}$, where $\Pr[\mathbf{G}(z) = 1] = W(z)$ independently for each $z \in \mathbb{F}_2^k$. Define $\mathbf{g}(x) = \mathbf{G}(\ell_1(x), \ldots, \ell_k(x))$, which is a randomized k-linear-junta and observe that $\mathbf{g}(x)$ and $h(x + \mathbf{y}_1 + \cdots + \mathbf{y}_N + \mathbf{v})$ have the same distribution for every $x \in \mathbb{F}_2^n$. Therefore, we have that

$$\Pr\left[\mathbf{g}(\mathbf{x}) = f(\mathbf{x})\right] = \Pr\left[h(\mathbf{x} + \mathbf{y}_1 + \ldots + \mathbf{y}_N + \mathbf{v}) = f(\mathbf{x})\right] \geq q - 2^{-\Omega(N)},$$

Finally, note that by an averaging argument, we can fix the internal randomness of $\mathbf{g}$ to obtain a k-linear-junta g so that

$$\Pr_{\mathbf{x} \sim D}\left[g(\mathbf{x}) = f(\mathbf{x})\right] \geq q - 2^{-\Omega(N)}.$$

This concludes the proof.

2.3 Sketching for three players

In this section we revise the proof of Theorem 8 and use a tool from additive combinatorics to obtain a version of Theorem 8 that requires only three players. In particular we show the following.

▶ **Theorem 12.** *Let $N \geq 3$ and suppose that F has a one-way broadcasting protocol with c bits of communication per message and success probability q. Then there exists a linear sketch of cost k which computes f with success probability $q - 0.01$, where $k = O(c^4)$.*

Note that the only caveat is that here we obtain $k = O(c^4)$ instead of the bound $k = O(c)$ in Theorem 8. The main technical lemma that we use is an almost periodicity result due to Croot and Sisask [13]. We quote the suitable version of the result bellow.

▶ **Lemma 13** (Theorem 3.2 of [28]). *Let $h' : \mathbb{F}_2^n \to \mathbb{R}$ be a function with $\|h'\|_\infty \leq 1$. Let $A_1, A_2 \subset \mathbb{F}_2^n$ and $|A_1|, |A_2| \geq \alpha |\mathbb{F}_2^n|$ for some $\alpha > 0$. Also let $\varepsilon > 0$ be an error parameter. Then there is a subspace V of codimension $O(\varepsilon^{-2} \log^4(\varepsilon^{-1}\alpha^{-1}))$ so that for all $x \in \mathbb{F}_2^n$,*

$$|h' * \varphi_{A_1} * \varphi_{A_2} * \varphi_V(x) - h' * \varphi_{A_1} * \varphi_{A_2}(x)| \leq \varepsilon.$$

Proof of Theorem 12. The proof is similar to the proof of Theorem 8 so we just explain the point where it differs. Find a transcript π and sets A_1, A_2 and functions $h : \mathbb{F}_2^n \to \{0,1\}$,$h' : \mathbb{F}_2^n \to \{-1,1\}$ as before that satisfy:

1. $[\boldsymbol{\pi} = \pi] \quad \Leftrightarrow \quad [\mathbf{x}_1 \in A_1, \mathbf{x}_2 \in A_2]$.
2. $|A_1|, |A_2| \geq 2^{-O(c)} 2^n$
3. $\Pr\left[h(\mathbf{x} + \mathbf{y}_1 + \mathbf{y}_2) = f(\mathbf{x})\right] = b(\pi) \geq q - 0.001$, for uniform and independent $\mathbf{y}_1 \in A_1, \mathbf{y}_2 \in A_2$.

Then instead of Claim 11, apply Lemma 13 to the function h', sets A_1, A_2, and $\varepsilon = 0.001$, to obtain the subspace V of codim $O(c^4)$ that we are looking for. Then continue the proof as before. ◀

3 Sketching for $f : \mathbb{F}_2^n \to [0,1]$

In this section, we approximate a given function $f : \mathbb{F}_2^n \to [0,1]$ with an additive error. Both the model and the proof are similar to the previous case.

3.1 Model

Protocols

The notions of *streaming protocol* and *one-way broadcasting protocol* with c bits of communication are defined similar as before, where the only difference is that the last message $\mathbf{m}_N$ takes values in $[0,1]$ instead of $\{0,1\}$.

The protocol G is said to *ε-approximate* $F : (\mathbb{F}_2^n)^N \to [0,1]$ if for all possible inputs $x_1, \ldots, x_N \in \mathbb{F}_2^n$, it holds that

$$\mathop{\mathbb{E}}_{\mathbf{r}} \left[|G(x_1, \ldots, x_N, \mathbf{r}) - F(x_1, \ldots, x_N)|^2 \right] \leq \varepsilon.$$

Linear sketches

A linear sketch of cost k is a distribution $\mathbf{g}$ over k-linear-juntas, where a k-linear-junta $g : \mathbb{F}_2^n \to [0,1]$ is defined as before. We think of $\mathbf{g}$ as a randomized function $\mathbf{g} : \mathbb{F}_2^n \to [0,1]$. The linear sketch $\mathbf{g}$ *ε-approximates* $f : \mathbb{F}_2^n \to [0,1]$ if, for every $x \in \mathbb{F}_2^n$, it holds that

$$\mathbb{E} \left[|\mathbf{g}(x) - f(x)|^2 \right] \leq \varepsilon.$$

We prove the following theorem in the rest of the section.

▶ **Theorem 14.** *Let $f : \mathbb{F}_2^n \to [0,1]$ and assume $N \geq 10n$ and $F : (\mathbb{F}_2^n)^N \to [0,1]$ is defined by $F(x_1, \ldots, x_N) = f(x_1 + \ldots + x_N)$. Suppose that F is ε-approximated by a one-way broadcasting protocol with c bits of communication per message. Then there is a linear sketch $\mathbf{g} : \mathbb{F}_2^n \to [0,1]$ of cost k that ε'-approximates f, where $k = O(c)$ and $\varepsilon' = 2\varepsilon + 2^{-\Omega(N)}$.*

▶ Remark 15. In this case as well, the proof directly generalizes to the case that $f : \mathbb{F}_p^n \to [0,1]$ for prime p conditioned on $N \geq 10n \log p$.

3.2 Proof of Theorem 14

The proof is similar to the proof of Theorem 8. We point out the necessary modifications. Same as before, we fix an arbitrary distribution D over $\mathbb{F}_2^n$, and show that there exists a k-linear-junta $g : \mathbb{F}_2^n \to [0,1]$ such that

$$\mathop{\mathbb{E}}_{\mathbf{x} \sim D} \left[|g(\mathbf{x}) - f(\mathbf{x})|^2 \right] \leq \varepsilon'.$$

To do so, obtain a function $h : \mathbb{F}_2^n \to [0,1]$ and sets $A_1, \cdots, A_N$ as before, so that for $\mathbf{x} \sim D, \mathbf{y}_i \in A_i$, we have

$$\mathbb{E} \left[|h(\mathbf{x} + \mathbf{y}_1 + \ldots + \mathbf{y}_N) - f(\mathbf{x})|^2 \right] \leq \varepsilon + 2^{-N}. \tag{5}$$

Now we switch to the exponential basis. Define functions $f', h' : \mathbb{F}_2^n \to \mathbb{C}$ by

$$f'(x) = e^{if(x)}, h'(x) = e^{-ih(x)}.$$

where $e = 2.71828\cdots$ is Euler's constant. Note that if $f(x) = h(x)$, then $\mathrm{Re}\,[f'(x)h'(x)] = 1$, where $\mathrm{Re}[z]$ is the real component of z. We need the following claim.

▷ Claim 16. Let $\mathbf{z}$ be a $[-1,1]$-valued random variable. Then,

$$1-\frac{1}{2}\mathbb{E}\left[\mathbf{z}^2\right] \le \mathbb{E}\left[\mathrm{Re}[e^{i\mathbf{z}}]\right] \le 1-\frac{1}{3}\mathbb{E}\left[\mathbf{z}^2\right].$$

Proof. The Taylor expansion of $\mathrm{Re}\, e^{iz} = cos(z)$ is $1-\frac{z^2}{2!}+\frac{z^4}{4!}-\cdots$. Therefore

$$1-\frac{z^2}{2} \le \mathrm{Re}[e^{iz}] \le 1-\frac{z^2}{3}$$

provided that $z \in [-1,1]$. ◁

Using the lower bound in Claim 16, by taking $\mathbf{z} = h(\mathbf{x}+\mathbf{y}_1+\ldots+\mathbf{y}_N)-f(\mathbf{x})$, we get

$$\mathbb{E}\left[\mathrm{Re}[h'(\mathbf{x}+\mathbf{y}_1+\ldots+\mathbf{y}_N)f'(\mathbf{x})]\right] \ge 1-\varepsilon/2-2^{-\Omega(N)}$$

We apply Claim 11 same as before to obtain subspaces V, U and

$$\mathbb{E}\left[\mathrm{Re}[h'(\mathbf{x}+\mathbf{y}_1+\ldots+\mathbf{y}_N+\mathbf{v})f'(\mathbf{x})]\right] \ge 1-\varepsilon/2-2^{-\Omega(N)} \tag{6}$$

Define a randomized k-linear-junta $\mathbf{r}:\mathbb{F}_2^n \to [0,1]$ as follows. Sample $\mathbf{y}_1,\cdots,\mathbf{y}_N,\mathbf{v}$. Then for every $u \in U$ and $v \in V+u$, set

$$\mathbf{r}(v) = h(u+\mathbf{y}_1+\cdots+\mathbf{y}_N+\mathbf{v}).$$

Observe that for every $x \in \mathbb{F}_2^n$, the randomized functions $\mathbf{r}(x)$ and $h(x+\mathbf{y}_1+\ldots+\mathbf{y}_N+\mathbf{v})$ have identical distributions, and therefore, $e^{-\pi i\mathbf{r}(x)}$ has the same distribution as $h'(x+\mathbf{y}_1+\cdots+\mathbf{y}_N+\mathbf{v})$. Combining this with Equation (6) implies

$$\mathbb{E}\left[\mathrm{Re}[e^{-\pi i\mathbf{r}(\mathbf{x})}f'(\mathbf{x})]\right] \ge 1-\varepsilon/2-2^{-\Omega(N)}$$

By an averaging argument, there is a k-linear-junta $g:\mathbb{F}_2^n \to [0,1]$ that

$$\mathbb{E}\left[\mathrm{Re}[e^{-\pi ig(\mathbf{x})}f'(\mathbf{x})]\right] \ge 1-\varepsilon/2-2^{-\Omega(N)}.$$

Finally, by using the upper bound in Claim 16, we get that

$$\mathbb{E}\left[|g(\mathbf{x})-f(\mathbf{x})|^2\right] \le 1-4\,\mathbb{E}\left[\mathrm{Re}[e^{-\pi ig(\mathbf{x})}f'(\mathbf{x})]\right] \le 2\varepsilon+2^{-\Omega(N)}$$

which finishes the proof.

4 Sketching over abelian groups of bounded exponent

Let G be a finite abelian group. We generalize Theorem 8 and Theorem 14 to the case where $f:G\to\{0,1\}$ and $f:G\to[0,1]$, respectively. Even though we are mostly interested in $G=\mathbb{Z}_p^n$, it turns out to be useful to consider this more general formulation. In particular, the proofs of Theorem 8 and Theorem 14 directly generalize to the case of prime p, but the case of composite p requires a more careful analysis. We introduce the required modifications to the definitions and the proofs.

Protocols

The concept of broadcasting protocol and streaming protocol for the function $F(x_1,\cdots,x_N) = f(x_1+\cdots+x_N)$ is defined as before.

Sketching

We modify the previous definition of sketching so that it will be meaningful for arbitrary abelian groups. Let H be an arbitrary subgroup of G. Let $Q = G/H$ be the quotient group. A function $g : G \to \{0,1\}$ is *H-invariant* if it is constant on every coset $H + w$ for all $w \in Q$. Note that such g can be factored as $g(x) = h(q(x))$ where $q : H \to Q$ is defined by $q(x) = x + H$ and $h : Q \to \{0,1\}$ is an arbitrary function. A function $g : G \to \{0,1\}$ has *linear complexity* r if there is a subgroup $H \leq G$ so that g is H-invariant and also $|G/H| \leq r$. Note that for functions $g : \mathbb{F}_2^n \to \{0,1\}$, the notion of k-linear-junta is equivalent to linear complexity 2^k.

A *linear sketch* of *complexity* r is a distribution $\mathbf{g} : G \to \{0,1\}$ over functions $g : G \to \{0,1\}$ of linear complexity r. We have the following two theorems for functions $g : G \to \{0,1\}$ and $g : G \to [0,1]$.

Simulation theorems

Before stating the theorems, we need to introduce one parameter of the group G that will be important here. Let the *exponent* of G, be the smallest m so that $m \cdot g = 0$ for all $g \in G$. Now, we can state the simulation theorems. Same as before, let $f : G \to \{0,1\}$ (respectively, $f : G \to [0,1]$) and define $F : G^N \to \{0,1\}$ (respectively, $F : G^N \to [0,1]$) by $F(x_1, \cdots, x_N) = f(x_1 + \cdots + x_N)$.

▶ **Theorem 17.** *Let G be an abelian group of exponent m. Let $N \geq 10n \log m$ and suppose that F has a one-way broadcasting protocol with c bits of communication per message and success probability q. Then there exists a linear sketch of complexity r which computes f with success probability $q - 2^{-\Omega(N)}$, where $r = m^{O(c)}$.*

And similarly, for bounded real-valued functions we have the following.

▶ **Theorem 18.** *Let G be an abelian group of exponent m and $N \geq 10n \log m$. Suppose that F is ε-approximated by a one-way broadcasting protocol with c bits of communication per message. Then there is a linear sketch $\mathbf{g} : \mathbb{F}_2^n \to [0,1]$ of complexity r that ε'-approximates f, where $r = m^{O(c)}$ and $\varepsilon' = 2\varepsilon + 2^{-\Omega(N)}$.*

We need to provide suitable versions of Lemma 10 and Claim 11 here. The other parts of the proof are same as before. We first have to introduce some notation to do Fourier analysis.

Fourier analysis

A character $\gamma : G \to \mathbb{C}^*$ of G is a homomorphism to the group $\mathbb{C}^*$. That is, for every $x, y \in G$ we have $\gamma(x+y) = \gamma(x)\gamma(y)$ and $\gamma(0) = 1$. The dual group of G, denoted by $\widehat{G}$, is the group of all characters of G. $\widehat{G}$ has the group structure introduced by $(\gamma_1 + \gamma_2)(x) = \gamma_1(x)\gamma_2(x)$. In fact $\widehat{G}$ is isomorphic to G. Given any function $f : G \to \mathbb{C}$, we can write f in its Fourier basis as

$$f(x) = \sum_{\gamma \in \widehat{G}} \widehat{f}(\gamma)\gamma(x)$$

where

$$\widehat{f}(\gamma) = \mathop{\mathbb{E}}_{x \in G} f(x)\overline{\gamma(x)}$$

and $\overline{\gamma(x)}$ is the complex conjugate of $\gamma(x)$. Moreover the convolution operator is defined as before. Given $f, g : G \to \mathbb{C}$, write $f * g(x) = \mathbb{E}_{y \in G} f(y)g(x-y)$ for $x \in G$, which leads to $\widehat{f * g}(\gamma) = \widehat{f}(\gamma)\widehat{g}(\gamma)$ for all $\gamma \in \widehat{G}$. Again, given a set $A \subset G$, define its normalized function as $\varphi_A = \frac{|G|}{|A|} 1_A$. We need two more definitions. Let $\Gamma \subset \widehat{G}$. Then $\Gamma^\perp$, called the *annihilator* of Γ is a subgroup of G defined by $\Gamma^\perp = \{x \in G : \gamma(x) = 1, \forall \gamma \in \Gamma\}$. A subset $\Gamma \subset \widehat{G}$ is called *dissociated* if there are no non-trivial solutions to the equation

$$\sum_{\gamma \in \Gamma} a_\gamma \cdot \gamma = 0$$

where each $a_\gamma \in \{-1, 0, 1\}$, $1 \cdot \gamma = \gamma$, $(-1) \cdot \gamma = -\gamma$, and $0 \cdot \gamma = 0$. Let us restate the general form of Chang's lemma [10].

▶ **Lemma 19.** *Let $A \subset G$ be a set of density $\alpha > 0$. Suppose that $\Gamma \subset \widehat{G}$ is a dissociated set. Then*

$$\sum_{\gamma \in \Gamma} |\widehat{\varphi_A}(\gamma)|^2 \leq O(\log \alpha^{-1}).$$

Note that if $\Gamma \subset \widehat{G}$ and $\Gamma' \subset \Gamma$ is the largest dissociated subset of Γ, then $\Gamma \subset \langle \Gamma' \rangle$, the span of Γ'. Since G has exponent m, we have $|\Gamma| \leq m^{|\Gamma'|}$. Moreover, one can show that $\Gamma^\perp \cong G/\langle \Gamma \rangle$, therefore, $|\langle \Gamma^\perp \rangle| \geq \frac{|G|}{m^{|\Gamma'|}}$.

Finally, the last part to modify in the proof of Theorems 17 and 18 is to obtain a suitable version of Claim 11. Using Chang's lemma as stated above and an analogous proof as before, we can find the function $h' : G \to \mathbb{C}$ taking values in the unit circle, and also the sets $A_1, \cdots, A_N \subset G$ as discussed in the proof of Theorem 8. Also using Lemma 19 we can find a maximal dissociated subset Γ' of size $k = O(c)$. Then take the subgroup $H = \Gamma'^\perp \leq G$ (as the analog of the subspace $V \leq \mathbb{F}_2^n$) so that $|G/H| \leq m^k$. The following claim about H is what we need.

▷ Claim 20. Let $\mathbf{y}_1 \in A_1, \ldots, \mathbf{y}_N \in A_N, \mathbf{v} \in H$ be chosen uniformly and independently. Then for every $x \in G$ it holds that

$$|\mathbb{E}[h'(x + \mathbf{y}_1 + \ldots + \mathbf{y}_N)] - \mathbb{E}[h'(x + \mathbf{y}_1 + \ldots + \mathbf{y}_N + \mathbf{v})]| \leq 2^{-N/8}|G|.$$

The proof is analogous to the proof of Claim 11. By taking $N \geq 10n \log m$ we can make sure that $2^{-N/8}|G| \leq 2^{-\Omega(N)}$. This finishes the proofs of Theorems 17 and 18.

References

1 Kook Jin Ahn, Sudipto Guha, and Andrew McGregor. Analyzing graph structure via linear measurements. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*, pages 459–467, 2012.

2 Kook Jin Ahn, Sudipto Guha, and Andrew McGregor. Graph sketches: sparsification, spanners, and subgraphs. In *Proceedings of the 31st ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2012, Scottsdale, AZ, USA, May 20-24, 2012*, pages 5–14, 2012.

3 Yuqing Ai, Wei Hu, Yi Li, and David P. Woodruff. New Characterizations in Turnstile Streams with Applications. In Ran Raz, editor, *31st Conference on Computational Complexity (CCC 2016)*, volume 50 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 20:1–20:22, Dagstuhl, Germany, 2016. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.CCC.2016.20.

4 Noga Alon, Yossi Matias, and Mario Szegedy. The Space Complexity of Approximating the Frequency Moments. *J. Comput. Syst. Sci.*, 58(1):137–147, 1999. doi:10.1006/jcss.1997.1545.

5 Sepehr Assadi, Sanjeev Khanna, and Yang Li. On Estimating Maximum Matching Size in Graph Streams. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 1723–1742, 2017.

6 Sepehr Assadi, Sanjeev Khanna, Yang Li, and Grigory Yaroslavtsev. Maximum Matchings in Dynamic Graph Streams and the Simultaneous Communication Model. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1345–1364, 2016. doi:10.1137/1.9781611974331.ch93.

7 László Babai, Anna Gál, Peter G. Kimmel, and Satyanarayana V. Lokam. Communication Complexity of Simultaneous Messages. *SIAM J. Comput.*, 33(1):137–166, 2003. doi:10.1137/S0097539700375944.

8 László Babai and Peter G. Kimmel. Randomized Simultaneous Messages: Solution of a Problem of Yao in Communication Complexity. In *Proceedings of the Twelfth Annual IEEE Conference on Computational Complexity, Ulm, Germany, June 24-27, 1997*, pages 239–246, 1997.

9 Sayan Bhattacharya, Monika Henzinger, Danupon Nanongkai, and Charalampos E. Tsourakakis. Space- and Time-Efficient Algorithm for Maintaining Dense Subgraphs on One-Pass Dynamic Streams. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 173–182, 2015.

10 Mei-Chu Chang et al. A polynomial bound in Freiman's theorem. *Duke mathematical journal*, 113(3):399–419, 2002.

11 Rajesh Chitnis, Graham Cormode, Hossein Esfandiari, MohammadTaghi Hajiaghayi, Andrew McGregor, Morteza Monemizadeh, and Sofya Vorotnikova. Kernelization via Sampling with Applications to Finding Matchings and Related Problems in Dynamic Graph Streams. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1326–1344, 2016.

12 Rajesh Hemant Chitnis, Graham Cormode, Mohammad Taghi Hajiaghayi, and Morteza Monemizadeh. Parameterized Streaming: Maximal Matching and Vertex Cover. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 1234–1251, 2015.

13 Ernie Croot and Olof Sisask. A probabilistic technique for finding almost-periods of convolutions. *Geometric and functional analysis*, 20(6):1367–1396, 2010.

14 Hossein Esfandiari, MohammadTaghi Hajiaghayi, and David P. Woodruff. Brief Announcement: Applications of Uniform Sampling: Densest Subgraph and Beyond. In *Proceedings of the 28th ACM Symposium on Parallelism in Algorithms and Architectures, SPAA 2016, Asilomar State Beach/Pacific Grove, CA, USA, July 11-13, 2016*, pages 397–399, 2016.

15 Martin Farach-Colton and Meng-Tsung Tsai. Tight Approximations of Degeneracy in Large Graphs. In *LATIN 2016: Theoretical Informatics - 12th Latin American Symposium, Ensenada, Mexico, April 11-15, 2016, Proceedings*, pages 429–440, 2016.

16 Sumit Ganguly. Lower Bounds on Frequency Estimation of Data Streams (Extended Abstract). In *Computer Science - Theory and Applications, Third International Computer Science Symposium in Russia, CSR 2008, Moscow, Russia, June 7-12, 2008, Proceedings*, pages 204–215, 2008.

17 Hamed Hatami, Kaave Hosseini, and Shachar Lovett. Structure of Protocols for XOR Functions. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 282–288, 2016. doi:10.1109/FOCS.2016.38.

18 Piotr Indyk. Stable distributions, pseudorandom generators, embeddings, and data stream computation. *J. ACM*, 53(3):307–323, 2006. doi:10.1145/1147954.1147955.

19 William B. Johnson and Joram Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. In *Conference in modern analysis and probability*, pages 189–206, 1984.

20 Sampath Kannan, Elchanan Mossel, Swagato Sanyal, and Grigory Yaroslavtsev. Linear Sketching over $\mathbb{F}_2$. In *33rd Computational Complexity Conference, CCC 2018, June 22-24, 2018, San Diego, CA, USA*, pages 8:1–8:37, 2018.

21 Michael Kapralov, Jelani Nelson, Jakub Pachocki, Zhengyu Wang, David P. Woodruff, and Mobin Yahyazadeh. Optimal Lower Bounds for Universal Relation, and for Samplers and Finding Duplicates in Streams. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 475–486, 2017. doi:10.1109/FOCS.2017.50.

22 Christian Konrad. Maximum Matching in Turnstile Streams. In *Algorithms - ESA 2015 - 23rd Annual European Symposium, Patras, Greece, September 14-16, 2015, Proceedings*, pages 840–852, 2015.

23 Yi Li, Huy L. Nguyen, and David P. Woodruff. Turnstile streaming algorithms might as well be linear sketches. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 174–183, 2014. doi:10.1145/2591796.2591812.

24 Andrew McGregor. Graph stream algorithms: a survey. *SIGMOD Record*, 43(1):9–20, 2014. doi:10.1145/2627692.2627694.

25 Andrew McGregor, David Tench, Sofya Vorotnikova, and Hoa T. Vu. Densest Subgraph in Dynamic Graph Streams. In *Mathematical Foundations of Computer Science 2015 - 40th International Symposium, MFCS 2015, Milan, Italy, August 24-28, 2015, Proceedings, Part II*, pages 472–482, 2015.

26 Ashley Montanaro and Tobias Osborne. On the communication complexity of XOR functions. *CoRR*, abs/0909.3392, 2009. arXiv:0909.3392.

27 Noam Nisan. Psuedorandom Generators for Space-Bounded Computation. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA*, pages 204–212, 1990.

28 Tomasz Schoen and Olof Sisask. Roth theorem for four variables and additive structures in sums of sparse sets. In *Forum of Mathematics, Sigma*, volume 4. Cambridge University Press, 2016.

29 Yaoyun Shi and Zhiqiang Zhang. Communication complexities of symmetric XOR functions. *Quantum Inf. Comput*, pages 0808–1762, 2008.

30 Justin Thaler. Semi-Streaming Algorithms for Annotated Graph Streams. In *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, pages 59:1–59:14, 2016.

31 Hing Yin Tsang, Chung Hoi Wong, Ning Xie, and Shengyu Zhang. Fourier Sparsity, Spectral Norm, and the Log-Rank Conjecture. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 658–667, 2013. doi:10.1109/FOCS.2013.76.

32 David P. Woodruff. Sketching as a Tool for Numerical Linear Algebra. *Foundations and Trends in Theoretical Computer Science*, 10(1-2):1–157, 2014. doi:10.1561/0400000060.

33 Penghui Yao. Parity Decision Tree Complexity and 4-Party Communication Complexity of XOR-functions Are Polynomially Equivalent. *Chicago J. Theor. Comput. Sci.*, 2016, 2016. URL: http://cjtcs.cs.uchicago.edu/articles/2016/12/contents.html.

34 Grigory Yaroslavtsev and Samson Zhou. Approximate $\mathbb{F}_2$-Sketching of Valuation Functions. *In submission to ITCS'19, available at http://grigory.us/files/approx-linsketch.pdf.*, 2018.

A Pseudorandom generators

Below we describe a standard application of pseudorandom generators for space bounded computation by Nisan [27] in the streaming setting following presentation given by Indyk [18]. Such pseudorandom generators can be used to fool any Finite State Machine which uses only $O(S)$ space or $2^{O(S)}$ states. Since a sketch consisting of s numbers modulo p can only take p^s values we can think of such sketches as being Finite State Machines with $O(s \log p)$ space.

Assume that a Finite State Machine Q which uses $O(S)$ bits of space uses at most k blocks of random bits where each block is of length b. The generator $G\colon \{0,1\}^m \to (\{0,1\}^b)^k$ expands a small number m of uniformly random bits into kb bits which "look random" for Q. Formally, let $U(\{0,1\}^t)$ be a uniform distribution over $\{0,1\}^t$. For any discrete random variable let $D[X]$ be the distribution of X interpreted is a vector of probabilities. Let $Q(x)$ denote the state of Q after using the random bits sequence x. Then G is a pseudorandom generator *with parameter* $\epsilon > 0$ for a class $\mathcal{C}$ of Finite State Machines, if for every $Q \in \mathcal{C}$:

$$|D[Q_{x\sim U(\{0,1\}^{bk})}] - D[Q_{x\sim U(\{0,1\}^m)}(G(x))]|_1 \leq \epsilon,$$

where $|y|_1$ denotes the ℓ_1-norm of a vector y.

▶ **Theorem 21** ([27]). *There exists a pseudorandom generator G with parameter $\epsilon = 2^{-O(S)}$ for Finite State Machines using space $O(S)$ such that:*

- *G expands $O(S \log R)$ bits into $O(R)$ bits.*
- *G requires only $O(S)$ bits of storage (in addition to its random input bits)*
- *Any length-$O(S)$ block of $G(x)$ can be computed using $O(\log R)$ arithmetic operations on $O(S)$-bit words.*

Using the above results we can reduce the amount of randomness used by a linear sketch modulo p as follows. Consider any state $\mathcal{S}$ of the linear sketch of dimension s. From the above discussion it follows that this state can be represented using $O(s \log p)$ bits. When a streaming update to coordinate i arrives we need only $O(s \log p)$ bits to generate the i-th row of the linear sketch matrix so that we can add it to the linear sketch. However, in order to ensure consistency, i.e. to make sure that when the i-th row is generate again we get the same result, we still need a lot of memory. Solution to this issue due to [18] follows below.

First, assume that the streaming updates (i, δ_t) are coming in the non-decreasing order of i. In this case we do not have to store the rows of the linear sketch matrix as we can generate them on the fly. Indeed, after the i-th row is generated we can apply it to all updates which contain i since such updates arrive sequentially one after another. This gives an algorithm which uses only $O(s \log p)$ storage and $O(n)$ blocks of random bits of size $O(s \log p)$ each. Hence there exists a pseudorandom generator G which given a random seed of size $O(s \log p \log(n/\delta))$ expands it to a pseudorandom sequence using which instead of the rows the sketch matrix only results in a negligible probability of error. I.e. the resulting state of the sketch can still be used to estimate the value of the function f of interest.

The key observation is that for every fixed random seed the resulting state doesn't depend on the order of updates (i, δ_t) by the commutativity of addition. Hence one can use G even if the updates come in any order.

Equality Alone Does not Simulate Randomness

Arkadev Chattopadhyay
School of Technology and Computer Science, Tata Institute of Fundamental Research, Mumbai, India
arkadev.c@tifr.res.in

Shachar Lovett
Department of Computer Science and Engineering, University of California, San Diego, USA
slovett@ucsd.edu

Marc Vinyals
School of Technology and Computer Science, Tata Institute of Fundamental Research, Mumbai, India
marc.vinyals@tifr.res.in

Abstract

The canonical problem that gives an exponential separation between deterministic and randomized communication complexity in the classical two-party communication model is "Equality". In this work we show that even allowing access to an "Equality" oracle, deterministic protocols remain exponentially weaker than randomized ones. More precisely, we exhibit a total function on n bits with randomized one-sided communication complexity $O(\log n)$, but such that every deterministic protocol with access to "Equality" oracle needs $\Omega(n)$ cost to compute it.

Additionally we exhibit a natural and strict infinite hierarchy within BPP, starting with the class P^{EQ} at its bottom.

2012 ACM Subject Classification Theory of computation → Communication complexity

Keywords and phrases Communication lower bound, derandomization

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.14

Funding *Arkadev Chattopadhyay*: Part of the work done while visiting the Simons Institute for the Theory of Computing.
Shachar Lovett: Supported by NSF award 1614023.
Marc Vinyals: Supported by the Prof. R Narasimhan Foundation.

Acknowledgements We are grateful to Bruno Loff, Jaikumar Radhakrishnan and especially Suhail Sherif for helpful discussions in the early stages of this work. We thank Sagnik Mukhopadhyay and anonymous reviewers for providing useful feedback to an earlier version of this manuscript. We also thank the Simons Institute for the Theory of Computing at Berkeley where some of this work took place.

1 Introduction

A deterministic communication protocol in Yao's two-party model is a strategy for a collaborative game between two parties, Alice and Bob, each of whom receives an input and whose task is to compute a function while communicating as little as possible.

It has been known since the origins of communication complexity that randomized protocols, where the parties are given access to a source of randomness and are allowed to make errors with small probability, are strictly more powerful than deterministic protocols. The classic example is the Equality function over n-bit strings, which has a randomized protocol with $\mathrm{O}(\log n)$ bits of communication, while every deterministic protocol requires at least $n + 1$ bits [15].

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka; Article No. 14; pp. 14:1–14:11

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

An efficient protocol for Equality is obtained by using a fingerprinting technique: use the randomness source to obtain a fingerprint of the strings to be compared of length $O(\log n)$, exchange the fingerprints, and answer whether the fingerprints are equal.

A few more examples of functions where randomness is helpful are the 'Greater-Than' function [13], the sparse set disjointness problem [8], and the Hamming distance problem with a small threshold [16]. In all cases the fingerprinting technique is enough to efficiently solve the problems. Is fingerprinting all there is to randomized protocols?

To state this question in a formal way we consider a model of communication where the parties are given access to an oracle that solves the Equality problem and are charged a cost of one bit each time the parties call the oracle. The set of functions that can be computed by some protocol in this model with cost $\operatorname{polylog} n$ bits is called P^{EQ}. The set of functions that have randomized protocols of cost $\operatorname{polylog} n$ is called BPP. We overload notation and use P^{EQ} and BPP to refer to both the class of functions and the corresponding communication models respectively. The question then is whether every function that has a randomized protocol with c bits of communication, also has a P^{EQ} protocol with $\operatorname{poly}(c, \log n)$ bits of communication and oracle calls. In other words, is $\mathsf{P}^{\mathrm{EQ}} = \mathsf{BPP}$?

The P^{EQ} model was first considered in [3]. The knowledge about it until our work (for total functions, see discussion below) can be summarized as follows:

$$\mathsf{P} \subsetneq \mathsf{P}^{\mathrm{EQ}} \subseteq \mathsf{BPP}.$$

P^{EQ} is also strictly weaker than the P^{NP} model, since EQ calls can be simulated with an NP oracle but P^{EQ} cannot efficiently solve the coNP-complete set disjointness problem. It also is worth mentioning that giving access to an Equality oracle is equivalent to giving access to a Greater-Than oracle up to a logarithmic factor. The latter model was introduced as real communication by Krajíček [10], with a connection to proof complexity in mind, and later found further applications in the same area [5, 4].

Partial functions

There are many examples in the literature of *partial functions* that separate P^{EQ} from BPP. One such example is the gap Hamming distance problem with a large gap. Concretely, the problem is to distinguish between pairs of input strings whose Hamming distance is less than a 1/3-fraction and more than a 2/3-fraction. This can be solved with a randomized protocol with $O(1)$ bits that samples a position in the strings uniformly at random and answers whether the strings are the same at that position. On the other hand, this problem has cost $\Omega(n)$ in the P^{NP} model [14], and hence in the P^{EQ} model too.

A different example follows from the simulation theorem of [5], made explicit in [6], and it is to lift a (partial) function that exhibits an exponential gap between deterministic and randomized query complexity, say promised majority. To be more precise, we consider the majority function of n bits with the promise that the fraction of zeros is either less than 1/3 or more than 2/3, which can be computed with a randomized decision tree by querying the input at a constant number of randomly sampled points, but requires linearly many queries to be solved by a deterministic decision tree. If we compose this function with the indexing gadget with pointers of size $O(\log n)$ then we have a randomized protocol of cost $O(\log n)$ that evaluates a constant number of instances of the gadget, while the simulation theorem tells us that it requires real communication $\Omega(n \log n)$.

Total functions

The question about a separation between P^{EQ} and BPP for *total functions* requires a different approach. If one uses the same means as before, namely lifting theorems, then a quadratic separation follows for example from the pointer chasing function [2] composed with indexing. However, this is where the lifting from query complexity approach seems to end, since deterministic and randomized query complexity are known to be polynomially related for total functions [12]. Our main result is a non-lifted total function, which exhibits an exponential separation between P^{EQ} and randomized communication.

▶ **Definition 1.1.** *The integer inner product problem* $\mathrm{IIP}_{m,t}(x,y)$ *is defined as follows. The inputs are integer vectors* $x, y \in [-M, M]^t$ *where* $M = 2^m$*. The output is* 1 *if* $\langle x, y\rangle = 0$*, where the inner product is computed over the integers.*

We denote by IIP_t the family of functions $\mathrm{IIP}_{m,t}$ with fixed $t = \mathrm{O}(1)$ and growing m. Note that the input size of $\mathrm{IIP}_{m,t}$ is $n = (m+1)t$.

▶ **Theorem 1.2** (Main theorem, informal)**.** *For any* $t \geq 6$*, the total function* IIP_t *on* n *bits can be computed with* $\mathrm{O}(\log n)$ *bits of randomized communication but requires* $\Omega(n)$ *cost to be solved by* P^{EQ} *protocols.*

Once we settled that EQ is not enough to simulate BPP because P^{EQ} cannot efficiently solve IIP, the next natural candidate for an oracle A such that $\mathsf{P}^A = \mathsf{BPP}$ becomes IIP itself. However, we also show that for any fixed t, IIP_t is not enough to simulate BPP, and in fact the complexity classes defined by IIP oracles form a strict infinite hierarchy.

▶ **Theorem 1.3.** *There is an infinite sequence* $(t_i)_{i\in\mathbb{N}}$ *such that*

$$\mathsf{P} \subsetneq \mathsf{P}^{\mathrm{EQ}} \subsetneq \mathsf{P}^{\mathrm{IIP}_{t_1}} \subsetneq \cdots \subsetneq \mathsf{P}^{\mathrm{IIP}_{t_i}} \subsetneq \mathsf{P}^{\mathrm{IIP}_{t_{i+1}}} \subsetneq \cdots \subset \mathsf{BPP} \ .$$

2 Preliminaries

We assume familiarity with standard definitions in communication complexity, such as in [11]. The only somewhat non-standard definition we need is that of protocols with access to an oracle.

If A is a family of communication problems $A_N \colon \{0,1\}^N \times \{0,1\}^N \to \{0,1\}$ for $N \in \mathbb{N}$, then the parties involved in a P^A protocol communicate via an oracle for A. Informally, if the players hold inputs $(x,y) \in \{0,1\}^n \times \{0,1\}^n$, every message is a pair of inputs $(g_1(x), g_2(y)) \in \{0,1\}^N \times \{0,1\}^N$ for one of the functions A_N, where g_1 and g_2 have been agreed beforehand, and the output $A_N(x,y)$ is visible to both parties. We assume that A is nontrivial in the sense that it can simulate sending one-bit messages from each party to the other one. The cost of such a protocol is the number of bits the oracle outputs, and $\mathsf{P}^A(f)$ is the minimum over all protocols. In particular, P^{EQ} is a protocol with oracle access to the Equality oracle, and P^{GT} is a protocol with oracle access to the Greater-Than oracle, both of which are nontrivial.

Usually the cost of an oracle call is defined with an additional term logarithmic in the size of its inputs, since otherwise we could solve any function with a single call to a strong oracle such as set disjointness. The kind of oracles we consider are weak enough that we do not need any limits on the input size to prove lower bounds, hence we omit the additional term for simplicity.

In fact, in our analysis, after a call to the oracle we immediately partition the set of inputs compatible with the answer into a set of rectangles. This makes it convenient to work

with a stronger model where all the possible sets of answers are partitioned beforehand, and the oracle tells the players not only the answer to their query, but also which rectangle in the partition their input belongs to, at no extra cost.

Observe that calling a function A_N with inputs transformed by g_1 and g_2 is equivalent to calling a function $B = A_N \circ (g_1, g_2)$ and that the matrix of B can be obtained from the matrix of A_N by removing, duplicating, and permuting some rows or columns. Therefore we identify an oracle A with the smallest family of matrices $\mathcal{M}_A$ that contains all the communication matrices of the functions A_N, and is closed under removing, duplicating, and permuting rows or columns. To each matrix $M \in \mathcal{M}_A$ we associate a monochromatic rectangle partition $\mathcal{R}(M)$, i.e., a set of rectangles such that for each rectangle $R \in \mathcal{R}(M)$ the submatrix of M defined by R is an all-zeros or all-ones matrix. In general, there may be many such choices; a good choice will be crucial for our lower bound technique. The only requirement is that this partition is to monochromatic rectangles, and hence a refinement of the answer given by the oracle.

A P^A protocol to compute a function $f\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ is a tree where each node corresponds to a rectangle $R \subseteq \{0,1\}^n \times \{0,1\}^n$ of compatible inputs. Each internal node is associated with a matrix $M \in \mathcal{M}_A$ of the same dimensions as R, and has one child for each rectangle $R' \in \mathcal{R}(M)$. Upon reaching a node labelled by R the players move to the child R' that contains their input. Each leaf is labelled 0 or 1, and the label of a leaf R equals $f(x, y)$ for each $(x, y) \in R$.

Analogous to how one bit of deterministic communication induces a refined partition of the input space where each rectangle is split into two, one call to an oracle induces a refined partition where each rectangle R is replaced by the partition $\mathcal{R}(M(R))$ associated to a matrix $M(R) \in \mathcal{M}_A$ of the same size. This is, we start with a single rectangle $\mathcal{R}_0 = \{\{0,1\}^n \times \{0,1\}^n\}$, and after i calls to the oracle we have the partition $\mathcal{R}_i = \bigcup_{R \in \mathcal{R}_{i-1}} \mathcal{R}(M(R))$. If a protocol computes a function f after c calls, then the partition $\mathcal{R}_c$ applied to M_f yields a set of monochromatic rectangles.

3 A Lower Bound Technique for P with Oracle Access

The goal of this section is to develop a lower bound technique for P^{EQ}, and more generally for P with oracle access. The key property of EQ that we exploit is that, no matter how it is transformed by an oracle call, we can always partition the matrix of EQ into few rectangles so that a large area is monochromatic. More generally, if we denote the number of elements in a matrix M by $|M|$, we define the property as follows.

▶ **Definition 3.1.** *A family of Boolean matrices $\mathcal{M}$ has* ϵ-monochromatic rectangles *if every matrix $M \in \mathcal{M}$ contains a monochromatic rectangle – i.e., an all-zeros or all-ones submatrix – of size at least $\epsilon|M|$.*

We obtain our lower bounds by estimating the following complexity measure.

▶ **Definition 3.2.** *If $\mathcal{R}$ is a set of rectangles and $\eta \in (1/2, 1)$ is a real number, we denote the* η-area *of $\mathcal{R}$ by $p_\eta(\mathcal{R}) = \sum_{R_i \in \mathcal{R}} |R_i|^\eta$. The η-area of a matrix M is the minimum of $p_\eta(\mathcal{R})$ over all monochromatic partitions $\mathcal{R}$ of M.*

Observe that the η-area of a matrix M is bounded below by $|M|^\eta$, which is attained if and only if the matrix is monochromatic, and above by $|M|$, which corresponds to partitioning the matrix into singletons. In fact, partitioning into either rows or columns gives a better upper bound of $2|M|^{(1+\eta)/2}$ for any matrix, and it can be shown that the matrix of inner product modulo 2 attains this bound up to a constant factor.

The *relative η-area* of a matrix M is $q_\eta(M) = p_\eta(M)/|M|^\eta$. Note that $q_\eta(M) \geq 1$ with equality attained if and only if M is monochromatic. The relative η-area of a family of matrices is the maximum relative η-area over all matrices in the family.

▶ **Lemma 3.3.** *For any η such that $1/(1-\log_2(1-\epsilon)) < \eta < 1$ there exists a constant $\xi = \xi(\epsilon,\eta)$ such that every ϵ-monochromatic family of matrices $\mathcal{M}$ that is closed under taking submatrices satisfies $q_\eta(\mathcal{M}) \leq \xi$.*

Proof. We prove the lemma by induction over the size of the matrices in the family. This is clearly true for 1×1 matrices; otherwise consider a matrix $M \in \mathcal{M}$ of size $r = |M|$. By assumption M contains a monochromatic rectangle R_1 of size $r_1 \geq \epsilon r$, so we can partition M into R_1 and two non-monochromatic rectangles R_2 and R_3 of respective sizes r_2 and r_3. We then apply the induction hypothesis to each non-monochromatic rectangle, while noting that the η-area of R_1 is r_1^η:

$$\begin{aligned} p_\eta(M) &\leq r_1^\eta + p_\eta(R_2) + p_\eta(R_3) \\ &\leq r_1^\eta + \xi r_2^\eta + \xi r_3^\eta \\ &\leq (1+2\xi)\left(\frac{r_1 + \xi r_2 + \xi r_3}{1+2\xi}\right)^\eta \\ &= (1+2\xi)^{1-\eta}\left(r_1 + \xi r_2 + \xi r_3\right)^\eta \\ &\leq (1+2\xi)^{1-\eta}\big(\xi + (1-\xi)\epsilon\big)^\eta r^\eta \ . \end{aligned}$$

We can write $(1-\epsilon) = (2+\delta)^{1-1/\eta}$ with $\delta > 0$ by the assumption on η. Set $\alpha = (2+\delta/2)^{1-1/\eta}$ so that $\alpha > (1-\epsilon)$ and set $\xi = \max\{2/\delta, \epsilon/(\alpha-(1-\epsilon))\}$. Then we can bound

$$1 + 2\xi = \xi(2 + 1/\xi) \leq \xi(2+\delta/2) = \xi\alpha^{1/(1-1/\eta)}$$

and

$$\xi + (1-\xi)\epsilon = \xi(1 - \epsilon + \epsilon/\xi) \leq \xi\alpha$$

so that

$$p_\eta(M) \leq (1+2\xi)^{1-\eta}\big(\xi + (1-\xi)\epsilon\big)^\eta r^\eta \leq \xi r^\eta\big(\alpha^{-\eta}\alpha^\eta\big) = \xi r^\eta \ . \qquad \blacktriangleleft$$

For simplicity we can take $\eta = 1 - \epsilon > 1/(1-\log_2(1-\epsilon))$ whenever $0 < \epsilon < 1/2$.

▶ **Lemma 3.4.** *Assume that f is a function which has a P^A protocol with cost c. For any $\eta \in (0,1)$ the communication matrix of f has relative η-area $q_\eta(f) \leq (q_\eta(\mathcal{M}_A))^c$.*

Proof. First we associate to each matrix $M \in \mathcal{M}_A$ a partition $\mathcal{R}(M)$ with relative η-area at most $q = q_\eta(\mathcal{M}_A)$. Next, assume that we have a partition of the input space into rectangles $\mathcal{R}$ with η-area $p_\eta(\mathcal{R})$. For each rectangle $R_i \in \mathcal{R}$ choose a matrix $M_i \in \mathcal{M}_A$ of the same dimensions. We obtain a refined partition $\mathcal{R}'$ by replacing each rectangle R_i by $\mathcal{R}(M_i)$. We can bound the total η-area of $\mathcal{R}'$ by

$$p_\eta(\mathcal{R}') = \sum_{R_i \in \mathcal{R}} p_\eta(\mathcal{R}(M_i)) \leq \sum_{R_i \in \mathcal{R}} q \cdot |R_i|^\eta = q \cdot p_\eta(\mathcal{R}) \ .$$

As $\mathcal{R}, \mathcal{R}'$ are partitions of the same dimensions, their relative η-areas satisfy

$$q_\eta(\mathcal{R}') \leq q \cdot q_\eta(\mathcal{R}) \ .$$

To conclude the proof, let $\mathcal{R}_0, \ldots, \mathcal{R}_c$ denote the intermediate partitions induced by the protocol, where $\mathcal{R}_i$ is the partition obtained after the first i calls. Then $\mathcal{R}_0$ is the singleton partition, $\mathcal{R}_c$ is a monochromatic partition of M_f, and all partitions have the same dimensions. Thus $q_\eta(\mathcal{R}_0) = 1$ and $q_\eta(\mathcal{R}_i) \leq q \cdot q_\eta(\mathcal{R}_{i-1})$ for $i = 1, \ldots, c$. We conclude that $q_\eta(M_f) \leq q_\eta(\mathcal{R}_c) \leq q^c$ as claimed. ◀

The next lemma gives an easy to verify condition under which Lemma 3.4 can be applied.

▶ **Lemma 3.5.** *Fix $0 < \eta < 1$. Let A be an oracle with constant relative η-area and let f be an n-bit function with a corresponding $2^n \times 2^n$ communication matrix M. Assume that:*
1. *The number of entries i, j with $M_{i,j} = 1$ is $\alpha 2^{2n}$.*
2. *For any 1-monochromatic rectangle R in M it holds that $|R| \leq \beta 2^{2n}$.*

Then the communication complexity of f in P^A is $\Omega(\log(\alpha\beta^{\eta-1}))$.

Proof. Let $\mathcal{R}$ be a partition of $f^{-1}(1)$ with minimum η-area. Let $x_i = |R_i|/2^{2n}$ denote the density of each rectangle R_i. Then the following minimization problem lower bounds the η-area of $\mathcal{R}$:

$$p_\eta(\mathcal{R}) \geq 2^{2\eta n} \cdot \min_{\sum_i x_i = \alpha, 0 \leq x_i \leq \beta} \sum_i x_i^\eta \ .$$

The minimum of a concave function over a convex polytope is attained at a vertex, in this case any point with $\lfloor \alpha/\beta \rfloor$ coordinates equal to β, one coordinate equal to $\alpha - \lfloor \alpha/\beta \rfloor \beta$, and the rest equal to 0. Hence

$$p_\eta(\mathcal{R}) \geq 2^{2\eta n} \lfloor \alpha/\beta \rfloor \beta^\eta \ .$$

If f has a P^A protocol with cost c, then by Lemma 3.4

$$p_\eta(\mathcal{R}) \leq 2^{2\eta n} (q_\eta(\mathcal{M}_A))^c = 2^{2\eta n + \mathrm{O}(c)} \ .$$

Rearranging these gives $c \geq \Omega(\log(\alpha\beta^{\eta-1}))$ as claimed. ◀

3.1 An Improved Bound for Equality

Coming back to the particular case of P^{EQ}, it is not hard to prove that the $\mathcal{M}_{\mathrm{EQ}}$ family of matrices has 1/9-monochromatic rectangles, and hence Lemma 3.5 applies to EQ with $\eta = 8/9$. While this is already enough to separate P^{EQ} and BPP, some of our applications require a tighter bound on η.

To obtain a better bound it is convenient to consider instead the model of P^{GT}, where the players have oracle access to a Greater-Than oracle. Note that as an EQ oracle can be simulated by two calls to a GT oracle, the latter model is stronger.

We show that $\mathcal{M}_{\mathrm{GT}}$ has constant η-area for any $\eta > 1/2$. The matrix GT_N is *monotone*, in the sense that it satisfies $M_{i_1,j_1} \leq M_{i_2,j_2}$ for all pairs of entries such that $i_1 \leq i_2$ and $j_1 \leq j_2$, and duplicating or removing rows and columns preserves monotonicity. Therefore every matrix $M \in \mathcal{M}_{\mathrm{GT}}$ is (a permutation of) a monotone matrix.

▶ **Lemma 3.6.** *A monotone matrix M can be partitioned into four rectangles R_1, R_2, R_3, R_4, such that R_1, R_2 are monochromatic and $|R_1| + |R_2| \geq |R_3| + |R_4|$.*

Proof. Let a and b be the dimensions of the matrix M and assume without loss of generality that $a \geq b$. Let a_1 be the maximal number such that $M_{a_1,b_1} = 0$, with $b_1 = \lceil a_1 b/a \rfloor$. Then the rectangle $R_1 = [1, a_1] \times [1, b_1]$ is 0-monochromatic, while the rectangle $R_2 = [a_1 + 1, a] \times$

$[b_1+1, b]$ is 1-monochromatic. We define $R_3 = [1, a_1] \times [b_1+1, b]$ and $R_4 = [a_1+1, a] \times [1, b_1]$. To complete the proof let $a_2 = a - a_1$ and $b_2 = b - b_2$, and observe that if $a_1 > a_2$ then $b_1 \geq b_2$, while if $a_1 < a_2$ then $b_1 \leq b_2$. Therefore by the rearrangement inequality

$$|R_1| + |R_2| = a_1 b_1 + a_2 b_2 \geq a_1 b_2 + a_2 b_1 = |R_3| + |R_4| \ . \qquad \blacktriangleleft$$

We use this partition to prove a more refined version of Lemma 3.3.

▶ **Lemma 3.7.** *For any $1/2 < \eta < 1$ there exists a constant $\xi = \xi(\eta)$ such that $q_\eta(\mathcal{M}_{\mathrm{GT}}) \leq \xi$.*

Proof. The proof is analogous to that of Lemma 3.3, except that we use Lemma 3.6 to partition each matrix into two monochromatic rectangles R_1 and R_2, and two non-monochromatic rectangles R_3 and R_4. We then get a bound

$$\begin{aligned} p_\eta(M) &\leq r_1^\eta + r_2^\eta + p_\eta(R_3) + p_\eta(R_4) \\ &\leq r_1^\eta + r_2^\eta + \xi r_3^\eta + \xi r_4^\eta \\ &\leq (2+2\xi)\left(\frac{r_1 + r_2 + \xi r_3 + \xi r_4}{2+2\xi}\right)^\eta \\ &\leq (2+2\xi)\left(\frac{r_1 + r_2 + r_3 + r_4}{4}\right)^\eta \\ &= \xi r^\eta \end{aligned}$$

for $\xi = 1/(2^{2\eta-1} - 1)$. $\blacktriangleleft$

It follows that Lemma 3.5 holds for both EQ and GT with $1/2 < \eta < 1$.

4 Separation

We demonstrate the separation by considering the inner product function over the integers. We recall the definition from the introduction.

▶ **Definition 4.1.** *The integer inner product problem $\mathrm{IIP}_{m,t}(x, y)$ is defined as follows. The inputs are integer vectors $x, y \in [-M, M]^t$ where $M = 2^m$. The output is 1 if $\langle x, y \rangle = 0$, where the inner product is computed over the integers.*

We use n to denote the input length, where $n = (m+1)t$. We recall that we consider $t = \mathrm{O}(1)$ and growing m.

▶ **Lemma 4.2.** *There is a* coRP *protocol for $\mathrm{IIP}_{m,t}$ of cost $\mathrm{O}(t \log m)$.*

Proof. Consider the following protocol: sample a uniformly random prime q among the first $4m + 2\log t$ primes, compute $\langle x, y \rangle \pmod q$ by having Alice send t integers $x_i \pmod q$ to Bob, and accept if and only if $\langle x, y \rangle = 0 \pmod q$. The protocol uses $\mathrm{O}(t \log q) = \mathrm{O}(t \log m)$ bits of communication.

The protocol is always correct on 1-inputs. To see that it is correct on 0-inputs with probability at least $1/2$ we observe that the probability of failure is the probability of picking a prime q that divides $\langle x, y \rangle$. Since the number $\langle x, y \rangle$ is bounded by tM^2 in absolute value, it is divisible by at most $\log(tM^2) = 2m + \log t$ primes, and since we have $4m + 2\log t$ primes to choose from, the probability of failure is at most $1/2$. $\blacktriangleleft$

▶ **Lemma 4.3.** *If t is even then $\Pr_{x,y}[\mathrm{IIP}_{m,t}(x, y) = 1] = \Omega(1/tM^2)$.*

Proof. Write $x = (x', -x'')$ and $y = (y', y'')$ where $x', y', x'', y'' \in [-M, M]^{t/2}$, so that $\langle x, y\rangle = \langle x', y'\rangle - \langle x'', y''\rangle$. The distribution of $\langle x', y'\rangle$ and $\langle x'', y''\rangle$ are i.i.d. and take at most $\mathrm{O}(tM^2)$ possible values. So the collision probability is $\Omega(1/tM^2)$. ◀

▶ **Lemma 4.4.** *For any rectangle $R \subseteq \mathrm{IIP}^{-1}_{m,t}(1)$ we have $|R| \leq (4M)^t$.*

Proof. Let $A, B \subset [-M, M]^t$ such that $\langle x, y\rangle = 0$ for all $x \in A, y \in B$. Let p be a prime between $2M + 1$ and $4M$, and consider the problem modulo p. Note that we can injectively identify A, B with subsets of $\mathbb{F}_p^t$. Let V, W denote the linear subspaces of $\mathbb{F}_p^t$ spanned by A, B, respectively. Then $V \perp W$ and hence $|V||W| \leq p^t$. This implies that $|A||B| \leq p^t \leq (4M)^t$. ◀

▶ **Lemma 4.5.** *Any P^{EQ} protocol for $\mathrm{IIP}_{m,t}$ with even $t \geq 6$ has cost $\Omega(n)$.*

Proof. Apply Lemma 3.5 with $\eta = \frac{1}{2} + \frac{1}{100}$, $\alpha = \Omega(1/tM^2)$ as given by Lemma 4.3, and $\beta = (4M)^t/(2M+1)^{2t} \leq 1/M^t$ as given by Lemma 4.4. We obtain

$$\mathsf{P}^{\mathrm{EQ}}(\mathrm{IIP}_{m,t}) = \Omega(\log(\alpha\beta^{\eta-1})) = \Omega(\log(M^{t(1-\eta)-2}/t)) = \Omega(tm) = \Omega(n) \ . \qquad ◀$$

Theorem 1.2 follows immediately from Lemma 4.2 and Lemma 4.5.

A related example

We give a similar separation by the inner product function over polynomials. Let $\mathbb{F}_2[z]$ denote the ring of univariate polynomials over $\mathbb{F}_2$.

▶ **Definition 4.6.** *The polynomial inner product problem $\mathrm{PIP}_{m,t}(x, y)$ is defined as follows. The inputs x, y are t-tuples of polynomials in $\mathbb{F}_2[z]$, each of degree at most m. The output is 1 if $\langle x, y\rangle = 0$, where the inner product is computed over $\mathbb{F}_2[z]$.*

Note that also here the input size is $n = (m+1)t$. Again we consider large m and $t = \mathrm{O}(1)$.

▶ **Lemma 4.7.** *There is a coRP protocol for $\mathrm{PIP}_{m,t}$ of cost $\mathrm{O}(t \log m)$.*

Proof. Consider the following protocol. Alice and Bob interpret their polynomials as polynomials in $\mathbb{F}_q[z]$ with $q = 2^{\lceil \log m \rceil + 2}$. They sample a uniformly random point $z \in \mathbb{F}_q$ and compute $\langle x, y\rangle(z)$ by having Alice send the result of evaluating each of her polynomials at z. The protocol uses $\mathrm{O}(t \log q) = \mathrm{O}(t \log m)$ bits of communication.

The protocol is always correct on 1-inputs. To see that it is correct on 0-inputs with probability at least $1/2$ we observe that the probability of failure is the probability of picking a root of $\langle x, y\rangle$. Since the number of roots is at most $2m$ and we have $q \geq 4m$ points in $\mathbb{F}_q$ to choose from, the probability of failure is at most $1/2$. ◀

▶ **Lemma 4.8.** *Any P^{EQ} protocol for $\mathrm{PIP}_{m,t}$ with even $t \geq 6$ has cost $\Omega(n)$.*

The proof is analogous to that of Lemma 4.5. We can use Lemma 4.3 unchanged, and we adapt Lemma 4.4 by considering the inner product function over $\mathbb{F}_q$ with $q = 2^m$.

Set disjointness

Babai et al. [3] were the first who attempted to prove a strong lower bound on the cost of any P^{EQ} protocol solving DISJ, however their method only yielded lower bounds for one-way protocols. The subsequent breakthrough tight bound of $\Omega(n)$ by Kalyanasundaram and Schnitger [9] on the randomized complexity of DISJ yields an $\Omega(n/\log n)$ bound on the P^{EQ} cost of DISJ. Using the techniques developed here, we prove a simple tight lower bound of $\Omega(n)$ on the cost of P^{EQ} protocols for set disjointness that does not rely on lower bounds for BPP.

▶ **Lemma 4.9.** *Any* P^{EQ} *protocol for* DISJ *has cost* $\Omega(n)$.

Proof. Apply Lemma 3.5 with $\eta = \frac{1}{2} + \frac{1}{100}$, $\alpha = (3/4)^n$, and $\beta = 1/2^n$. We obtain

$$\mathsf{P}^{\mathrm{EQ}}(\mathrm{DISJ}) = \Omega(\log(\alpha\beta^{\eta-1})) = \Omega(\log(2^{(\log 3 - 2 + 0.49)n}) = \Omega(\log(2^{0.07n})) = \Omega(n) \ . \qquad \blacktriangleleft$$

5 Hierarchy

A generic way to obtain ϵ-monochromatic families is by extracting large rectangles from matrices of small *sign-rank*. A real matrix M, each of whose entries are non-zero, is said to be sign represented by another matrix A if each entry of A and M agree in sign. The sign-rank of M is the minimum r such that there exists an A of rank r that sign represents it. A corollary of the following theorem allows us to extract large rectangles from matrices of small sign-rank.

▶ **Theorem 5.1** ([1])**.** *Let* U *and* V *be finite multisets of vectors in* $\mathbb{R}^d$ *and let* $\delta = 1/2^{d+1}$. *Then there are subsets* $U' \subset U$ *and* $V' \subset V$ *such that* $|U'| \geq \delta|U|$, $|V'| \geq \delta|V|$, *and either* $\langle u, v\rangle \geq 0$ *for all* $u, v \in U' \times V'$ *or* $\langle u, v\rangle < 0$ *for all* $u, v \in U' \times V'$.

▶ **Corollary 5.2.** *A Boolean matrix of sign-rank* d *and size* r *contains a monochromatic rectangle of size at least* $1/2^{2(d+1)}r$.

Proof. Let M be a matrix of size $n \times m$ and sign rank d, and let A and B be matrices of size $n \times d$ and $d \times m$ such that $M = \mathrm{sign}(AB)$. Apply Theorem 5.1 to the set of rows of A and the set of columns of B. $\blacktriangleleft$

Since sign-rank does not increase with respect to removing, duplicating, or permuting rows or columns, in order to establish that IIP_t is ϵ-monochromatic, it is sufficient to look at the sign-rank of IIP_t.

▶ **Lemma 5.3.** *The sign-rank of* $\mathrm{IIP}_{m,t}$ *is at most* $t^2 + 1$.

Proof. $\mathrm{IIP}_{m,t}(x, y) = \mathrm{sign}(\langle x, y\rangle^2 - 1/2)$, which can be decomposed into a linear combination of t^2 rank-one matrices of the form $M_{x,y} = \langle x_i x_j, y_i y_j\rangle$ and the all-ones matrix. $\blacktriangleleft$

We can now put all the pieces together and prove a lower bound for $\mathsf{P}^{\mathrm{IIP}_t}$.

▶ **Lemma 5.4.** *Any* $\mathsf{P}^{\mathrm{IIP}_t}$ *protocol for* $\mathrm{IIP}_{m,t'}$ *with even* $t' \geq 2^{3t^2}$ *has cost* $\Omega(n)$.

Proof. Let $\epsilon = 1/2^{2(t^2+1)}$ given by Corollary 5.2. Then $\mathcal{M}_{\mathrm{IIP}_t}$ is an ϵ-monochromatic family, therefore we can apply Lemma 3.5 with $\eta = 1 - \epsilon$. Choose t' to be the smallest even integer such that $(2\log t')/t' < (1 - \eta)$. We can bound t' by

$$t' \leq \frac{4}{1-\eta}\log\left(\frac{1}{1-\eta}\right) = \frac{4}{\epsilon}\log\left(\frac{1}{\epsilon}\right) \leq 2^{3t^2} \ .$$

Apply Lemma 3.5 with $\alpha = \Omega(1/t'M^2)$ as given by Lemma 4.3, and $\beta \leq 1/M^{t'}$ as given by Lemma 4.4. We obtain

$$\mathsf{P}^{\mathrm{IIP}_t}(\mathrm{IIP}_{m,t'}) = \Omega(\log(\alpha\beta^{\eta-1})) = \Omega\left(t'm\left(-\frac{2\log t'}{t'} + 1 - \eta\right)\right) = \Omega(t'm) = \Omega(n) \ . \qquad \blacktriangleleft$$

To prove Theorem 1.3 we consider the sequence of classes $\mathsf{P}^{\mathrm{IIP}_{t_i}}$ where $t_1 = 6$ and $t_{i+1} = 2^{3t_i^2}$. The inclusion $\mathsf{P}^{\mathrm{EQ}} \subseteq \mathsf{P}^{\mathrm{IIP}_{t_1}}$ follows from the observation that $\mathrm{EQ}(x, y) = \mathrm{IIP}_2((x, 1), (-1, y))$, and Lemma 4.5 shows that it is strict. The inclusions $\mathsf{P}^{\mathrm{IIP}_{t_i}} \subseteq \mathsf{P}^{\mathrm{IIP}_{t_{i+1}}}$ are immediate since we can solve $\mathrm{IIP}_{m,t}$ with a single call to $\mathrm{IIP}_{m,t'}$ padding the additional coordinates with zeros, and we just proved the non-inclusions in Lemma 5.4.

6 Concluding Remarks

This work belongs to the general area of understanding the power of randomness in communication complexity. We use this opportunity to remind the readers of a fascinating open problem, posed explicitly by Göös, Pitassi and Watson [7], which is whether $\mathsf{BPP} \subset \mathsf{P}^{\mathsf{NP}}$ for *total functions*. It is known that this containment is not true for partial functions. Göös et al. suggested, as a first step, separating the class of total functions in BPP from an interesting subclass of P^{NP}. In this work, we took this step by providing the first (exponential) separation between BPP and P^{EQ}, the latter being one of the most natural subclasses of P^{NP}. However, the original problem of separating BPP from P^{NP} remains open.

To state this in combinatorial terms, a function f has a P^{NP} protocol of cost c if the following holds. There exists a list of 2^c rectangles R_i and values $z_i \in \{0,1\}$, such that $f(x,y) = z_i$ for the *first* rectangle R_i in the list for which $(x,y) \in R_i$ (We may assume that the last rectangle contains all possible inputs, to make this model well defined). In particular, if $\mathsf{BPP} \subset \mathsf{P}^{\mathsf{NP}}$ then there must exist a monochromatic rectangle in f of density $2^{-\mathrm{O}(c)}$ for $c = \operatorname{polylog} n$. Understanding this question seems to be pivotal towards understanding the relation between BPP and P^{NP}.

▶ **Problem 6.1.** *Let f be an n-bit total Boolean function with a randomized protocol of cost c. Is it true that f must contain a monochromatic rectangle R of size $|R| \geq 2^{-\mathrm{O}(c)}2^{2n}$?*

References

1. Noga Alon, János Pach, Rom Pinchasi, Radoš Radoičić, and Micha Sharir. Crossing patterns of semi-algebraic sets. *Journal of Combinatorial Theory, Series A*, 111(2):310–326, 2005. doi:10.1016/j.jcta.2004.12.008.
2. Andris Ambainis, Kaspars Balodis, Aleksandrs Belovs, Troy Lee, Miklos Santha, and Juris Smotrovs. Separations in Query Complexity Based on Pointer Functions. *Journal of the ACM*, 64(5):32:1–32:24, 2017. Preliminary version in *STOC '16*. doi:10.1145/3106234.
3. László Babai, Péter Frankl, and János Simon. Complexity Classes in Communication Complexity Theory. In *Proceedings of the 27th Annual Symposium on Foundations of Computer Science (FOCS '86)*, pages 337–347, October 1986. doi:10.1109/SFCS.1986.15.
4. Paul Beame, Noah Fleming, Russell Impagliazzo, Antonina Kolokolova, Denis Pankratov, Toniann Pitassi, and Robert Robere. Stabbing Planes. In *Proceedings of the 9th Innovations in Theoretical Computer Science Conference (ITCS '18)*, volume 94 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 10:1–10:20, January 2018. doi:10.4230/LIPIcs.ITCS.2018.10.
5. María Luisa Bonet, Juan Luis Esteban, Nicola Galesi, and Jan Johannsen. On the Relative Complexity of Resolution Refinements and Cutting Planes Proof Systems. *SIAM Journal on Computing*, 30(5):1462–1484, 2000. Preliminary version in *FOCS '98*. doi:10.1137/S0097539799352474.
6. Susanna F. de Rezende, Jakob Nordström, and Marc Vinyals. How Limited Interaction Hinders Real Communication (and What It Means for Proof and Circuit Complexity). In *Proceedings of the 57th Annual IEEE Symposium on Foundations of Computer Science (FOCS '16)*, pages 295–304, October 2016. doi:10.1109/FOCS.2016.40.
7. Mika Göös, Toniann Pitassi, and Thomas Watson. The Landscape of Communication Complexity Classes. *Computational Complexity*, 27(2):245–304, June 2018. Preliminary version in *ICALP '16*. doi:10.1007/s00037-018-0166-6.
8. Johan Håstad and Avi Wigderson. The Randomized Communication Complexity of Set Disjointness. *Theory of Computing*, 3(1):211–219, 2007. doi:10.4086/toc.2007.v003a011.
9. Bala Kalyanasundaram and Georg Schnitger. The Probabilistic Communication Complexity of Set Intersection. *SIAM J. Discrete Math.*, 5(4):545–557, 1992. doi:10.1137/0405044.

10 Jan Krajíček. Interpolation by a Game. *Mathematical Logic Quarterly*, 44(4):450–458, 1998. doi:10.1002/malq.19980440403.

11 Eyal Kushilevitz and Noam Nisan. *Communication Complexity*. Cambridge University Press, New York, NY, USA, 1997.

12 Noam Nisan. CREW PRAMs and decision trees. *SIAM Journal on Computing*, 20(6):999–1007, December 1991. doi:10.1137/0220062.

13 Noam Nisan. The communication complexity of threshold gates. In *Proceedings of Combinatorics, Paul Erdős is Eighty*, volume 1, pages 301–315, 1993.

14 Periklis A. Papakonstantinou, Dominik Scheder, and Hao Song. Overlays and Limited Memory Communication. In *Proceedings of the 29th IEEE Conference on Computational Complexity (CCC '14)*, pages 298–308, June 2014. doi:10.1109/CCC.2014.37.

15 Andrew Chi-Chih Yao. Some Complexity Questions Related to Distributive Computing (Preliminary Report). In *Proceedings of the 11th Annual ACM Symposium on Theory of Computing (STOC '79)*, pages 209–213, April 1979. doi:10.1145/800135.804414.

16 Andrew Chi-Chih Yao. On the power of quantum fingerprinting. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing (STOC '03)*, pages 77–81, June 2003. doi:10.1145/780542.780554.

Counting Basic-Irreducible Factors Mod p^k in Deterministic Poly-Time and p-Adic Applications

Ashish Dwivedi
CSE, Indian Institute of Technology, Kanpur, India
ashish@cse.iitk.ac.in

Rajat Mittal
CSE, Indian Institute of Technology, Kanpur, India
rmittal@cse.iitk.ac.in

Nitin Saxena
CSE, Indian Institute of Technology, Kanpur, India
nitin@cse.iitk.ac.in

Abstract

Finding an irreducible factor, of a polynomial $f(x)$ modulo a prime p, is not known to be in deterministic polynomial time. Though there is such a classical algorithm that *counts* the number of irreducible factors of $f \bmod p$. We can ask the same question modulo prime-powers p^k. The irreducible factors of $f \bmod p^k$ blow up exponentially in number; making it hard to describe them. Can we count those irreducible factors mod p^k that remain irreducible mod p? These are called *basic-irreducible*. A simple example is in $f = x^2 + px \bmod p^2$; it has p many basic-irreducible factors. Also note that, $x^2 + p \bmod p^2$ is irreducible but not basic-irreducible!

We give an algorithm to count the number of basic-irreducible factors of $f \bmod p^k$ in deterministic $\mathrm{poly}(\deg(f), k \log p)$-time. This solves the open questions posed in (Cheng et al, ANTS'18 & Kopp et al, Math.Comp.'19). In particular, we are counting roots mod p^k; which gives the first deterministic poly-time algorithm to compute Igusa zeta function of f. Also, our algorithm efficiently partitions the set of all basic-irreducible factors (possibly exponential) into merely $\deg(f)$-many disjoint sets, using a compact tree data structure and *split* ideals.

2012 ACM Subject Classification Computing methodologies → Representation of mathematical objects; Mathematics of computing → Discrete mathematics; Theory of computation → Algebraic complexity theory; Computing methodologies → Hybrid symbolic-numeric methods; Mathematics of computing → Combinatoric problems; Computing methodologies → Number theory algorithms

Keywords and phrases deterministic, root, counting, modulo, prime-power, tree, basic irreducible, unramified

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.15

Acknowledgements We thank Vishwas Bhargava for introducing us to the open problem of factoring $f \bmod p^3$ and the related prime-power questions. A.D. thanks Sumanta Ghosh for the discussions. N.S. thanks the funding support from DST (DST/SJF/MSA-01/2013-14). R.M. would like to thank support from DST through grant DST/INSPIRE/04/2014/001799. We thank anonymous reviewers for their helpful comments and suggestions to improve the introduction section of this paper.

1 Introduction

Factoring a univariate polynomial, over *prime* characteristic, is a highly well studied problem. Though efficient factoring has been achieved using randomization, still efficient derandomization is a longstanding problem. A related question of equal importance is root finding, but this is known to be equivalent to factoring in deterministic poly-time. Surprisingly, testing irreducibility, or even counting irreducible factors, is easy in this regime. The main tool here is the magical Frobenius morphism of prime p characteristic rings: $x \mapsto x^p$.

Though much effort has been put in prime characteristic, few results are known in *composite* characteristic n [27]. Even irreducibility testing of a polynomial, with the prime factorization of n given, has no efficient algorithm known. This reduces to *prime-power* characteristic p^k [32]. Deterministic factoring in such a ring is a much harder question (at least it subsumes deterministic factoring mod p). In fact, even randomized algorithms, or practical solutions, are currently elusive [31, 32, 17, 26, 30, 13]. The main obstruction is non-unique factorization.

Being a non-unique factorization domain, there could be exponential number of roots, or irreducible factors, modulo prime-powers [31]. So one could ask a related question about counting all the irreducible factors (respectively roots) modulo prime-powers. Efficiently solving this counting problem will give us an efficient irreducibility testing criteria, which is the first question one wants to try. Recall that prime characteristic allows such an efficient method.

Motivated by this, we ask – Could we describe all factors which remain irreducible mod p? Such factors are called *basic*-irreducible in the literature. This is much more than counting roots mod p^k (as, $f(\alpha) = 0$ iff $x - \alpha$ is a basic-irreducible factor of f). These roots, besides being naturally interesting, have various applications in – factoring [7, 8, 4], coding theory [2, 26], elliptic curve cryptography [20], arithmetic algebraic-geometry [35, 12, 11, 16]. Towards this we design a machinery, yielding the following result:

Given a degree d integral polynomial $f(x)$ and a prime-power p^k, we partition the set of all basic-irreducible factors of $f \bmod p^k$ into at most d (compactly provided) subsets in deterministic poly$(d, k \log p)$-time; in the same time we count the number of factors in each of these subsets.

Also, we can compactly partition (and count) the roots of $f \bmod p^k$ in deterministic poly-time.

This efficient partitioning of (possibly exponentially many) roots into merely d subsets is reminiscent of the age-old fact: there are at most $\deg(g)$ roots of a polynomial $g(x)$ over a field. Root sets mod p^k are curious objects; not every subset of $\mathbb{Z}/p^k\mathbb{Z}$ is a root set (except when $k = 1$). Their combinatorial properties have been studied extensively [29, 9, 3, 10, 22]. In this regard, our result is one more step to understand the hidden properties of root-sets mod prime-powers.

Factoring mod p^k has applications in factoring over *local* fields [7, 8, 4]. Previously, the latter was achieved through randomized factoring mod p [5] and going to extensions of $\mathbb{Q}_p$. Directly factoring mod p^k, for arbitrary k, would imply a new and more natural factoring algorithm over p-adic fields. In fact, *our method gives the first deterministic poly-time algorithm to count basic-irreducible factors of $f \in \mathbb{Q}_p[x]$*; by picking k such that $p^{k/2} \nmid \text{discriminant}(f)$ (see [32, Thm. 3.11]). This derandomization was not known before, though $\mathbb{Q}_p[x]$ is indeed a unique factorization domain.

1.1 Previously known results

The questions of root finding and root counting of $f \bmod p^k$ are of classical interest, see [25, 1]. Using Hensel lifting (Section A) we know how to "lift" a root, of multiplicity one, of $f \bmod p$ to a root of $f \bmod p^k$, in a unique way. But this method breaks down when the root (mod p) has multiplicity more than one. [2, Cor.4] was the first work to give an efficient randomized algorithm to count, as well as find, all the roots of $f \bmod p^k$. [24] improved the time complexity of [2]. In this line of progress, very recently [6] gave a deterministic algorithm to count roots in time *exponential* in the parameter k. Extending the idea of [6], [19] gave

another efficient randomized algorithm to count roots of $f \bmod p^k$. Note that finding the roots deterministically seems a difficult problem because it requires efficient deterministic factoring of $f \bmod p$ (which is a classical open problem). But counting the roots mod p^k deterministically may be an easier first step.

Recently there has been some progress in factoring $f \bmod p^k$ when k is constant. [13] gave the first efficient randomized algorithm to factor $f \bmod p^k$ for $k \leq 4$. This gives an exponential improvement over the previous best algorithms of [30, 32, 31] mod p^k ($k \leq 4$). In fact, they generalized Hensel lifting method to mod p^k, for $k \leq 4$, in the difficult case when $f \bmod p$ is power of an irreducible. The related derandomization questions are all open.

The case of factoring $f \bmod p^k$ when k is "large" – larger than the maximum power of p dividing the discriminant of the integral f – has an efficient randomized algorithm due to [32]. They showed, assuming large k, that factorization mod p^k is well behaved and corresponds to the unique p-adic factorization of f (i.e. in $\mathbb{Q}_p[x]$). In turn, p-adic factoring has known efficient randomized algorithms [7, 8, 4]. The derandomization questions are all open.

We now give a deterministic method to count all the roots (resp. basic-irreducible factors) efficiently. In fact, our proof can be seen as a deterministic poly-time reduction of basic-irreducible factor finding mod p^k to root finding mod p. In particular, it subsumes all the results of [2].

1.2 Our results

▶ **Theorem 1** (Root count). *Let p be a prime, $k \in \mathbb{N}$ and $f(x) \in \mathbb{Z}[x]$. Then, all the roots of $f \bmod p^k$ can be counted in deterministic poly($\deg f, k \log p$)-time.*

This is the first efficient derandomization of the randomized root counting algorithms [2, 19], and an exponential improvement over the recent deterministic algorithm of [6]. The challenge arises from the fact that we need to count the possibly exponentially many roots without being able to find them[1].

Remarks.

1) In the algorithm, the (possibly exponential) root-set of $f \bmod p^k$ gets partitioned into at most $\deg(f)$-many disjoint subsets and we output a compact representation, called *split ideal*, for each of these subsets. We do count them, but do not yet know how to find a root deterministically.
2) This gives an efficient way to deterministically compute the Igusa zeta function, given an integral univariate f and a prime p. This follows from the fact that we just need to compute $N_k(f)$:=number of roots of $f \bmod p^k$, for $k \in [\ell]$ s.t. $p^{\ell/2} \nmid \text{discriminant}(f)$, to estimate *Poincaré series* $\sum_{i=0}^{\infty} N_i(f)x^i$ [11, 16]. Interestingly, it converges to a rational function!

   The proof follows from the fact that for each $i > l$, $N_i(f)$ is the sum of $t \leq d$ many p-powers where t is constant for each $i > l$ (see [32, Thm. 3.11]). So the sum $\sum_{i>l}^{\infty} N_i(f)x^i$ converges.
3) This is the first deterministic poly-time algorithm to count the number of lifts of a *repeated* root of $f \bmod p$ to $f \bmod p^k$.

[1] Note that counting roots of a multivariate polynomial over a finite field is #P-complete, even if the degree is restricted to be three [14].

4) This gives the first deterministic poly-time algorithm to count the number of p-adic integral roots of a given p-adic polynomial $f \in \mathbb{Q}_p[x]$. (Count roots mod p^ℓ where $p^{\ell/2} \nmid \text{discriminant}(f)$ [32, Thm. 3.11].)

Next, we extend the ideas for counting roots to count all the basic-irreducible factors of $f \bmod p^k$ in deterministic polynomial time. Recall that a *basic-irreducible* factor of $f \bmod p^k$ is one that remains irreducible in mod p arithmetic.

▶ **Theorem 2** (Factor count). *Let p be a prime, $k \in \mathbb{N}$ and $f(x) \in \mathbb{Z}[x]$. Then, all the basic-irreducible factors of $f \bmod p^k$ can be counted in deterministic poly($\deg f, k \log p$)-time.*

We achieve this by extending the idea of counting roots to more general p-adic integers. Essentially, we efficiently count all the roots of $f(x)$ in $\mathcal{O}_K/\langle p^k \rangle$, where $\mathcal{O}_K$ is the ring of integers of a p-adic *unramified* extension $K/\mathbb{Q}_p$ (refer [18] for the standard notation). Currently, there is no fast, practical method known to find/count roots when K is *ramified*.

▶ **Corollary 3.** *Consider (an unknown) p-adic extension $K := \mathbb{Q}_p[y]/\langle g(y) \rangle$, which is unramified and has degree Δ. Let $f(x) \in \mathbb{Z}[x]$, p, k, Δ be given as input (in binary).*

Then, we can count all the roots of f, in $\mathcal{O}_K/\langle p^k \rangle$, in deterministic poly($\deg(f), k \log p, \Delta$)-time.

Remarks.

1) This gives the first deterministic poly-time algorithm to count the number of (unramified p-adic integral) roots of a given p-adic polynomial $f \in K[x]$.
2) Our method generalizes to efficiently count all the roots of a given polynomial $f(x) \in (\mathbb{F}[t]/\langle h(t)^k \rangle)[x]$ for a given polynomial h (resp. $f \in \mathbb{F}[[t]][x]$ with power-series coefficients); assuming that $\mathbb{F}$ is a field over which root counting is efficient (eg. $\mathbb{Q}, \mathbb{R}, \mathbb{F}_p$ and their algebraic extensions).

1.3 Proof techniques

Our implementation involves constructing a *list data structure* $\mathcal{L}$ which implicitly partitions the root-set of $f \bmod p^k$ into at most $\deg(f)$-many disjoint subsets; and count the number of roots in each such subset. The construction of $\mathcal{L}$ is incremental, by doing arithmetic modulo special ideals,

Split ideals. A *split ideal* I_l of length $l+1$, and degree b, is a "triangular" ideal defined as $I_l = \langle h_0(x_0), h_1(\bar{x}_1), \ldots, h_l(\bar{x}_l) \rangle$, where the notation $\bar{x}_i$ refers to the variable set $\{x_0, \ldots, x_i\}$ and $b = \prod_{0 \le i \le l} \deg_{x_i}(h_i)$. It implicitly stores a size-b subset of the root-set of $f \bmod p^k$, where a root looks like $\sum_{0 \le i \le l} x_i p^i$ till precision p^{l+1}. Note that a root r of $f \bmod p^k$ is also a root of $f \bmod p^l$ for all $l \in [k]$. Since we cannot access them directly, we "virtualize" them in the notation $\bar{x}_l$.

The structure of these ideals is quite nice and recursive (Section 2). So it may keep splitting (in Algorithm 1) till it becomes a *maximal ideal*, which corresponds to a single point in $(\mathbb{F}_p)^l$ and has degree one. Or, the algorithm may halt earlier, due to "stable clustering" of roots, and then we call the ideals– *maximal split ideal*; in fact, $\mathcal{L}$ has only maximal split ideals. These do not give us the actual roots but do give us their count!

List data structure. $\mathcal{L}$ implicitly stores, and may partition, the root-set of $f \bmod p^k$. Essentially, $\mathcal{L}$ is a set of at most d maximal split ideals, i.e. $\mathcal{L} = \{I_1(l_1, d_1), \ldots, I_n(l_n, d_n)\}$, where each ideal $I_j \subseteq \mathbb{F}_p[\bar{x}_{k-1}]$ has two parameters – length l_j and degree d_j. A maximal split ideal $I(l, D)$ implicitly stores a size-D subset of the root-set of $f \bmod p^k$. This yields a simple count of Dp^{k-l} for the corresponding roots. Ideals in $\mathcal{L}$ have the property that they represent disjoint subsets of roots; and they collectively represent the whole root-set of $f \bmod p^k$. Thus, $\mathcal{L}$ gives us both the (implicit) structure and the (exact) size of the root-set of $f \bmod p^k$. In the intermediate steps of the algorithm, for efficiency reasons, we will store a tuple (I_j, f_{I_j}) in a changing *stack* S. Where, $f_{I_j}(\bar{x}_{l_j-1}, x) := f(x_0 + px_1 + \cdots + p^{l_j-1}x_{l_j-1} + p^{l_j}x) \bmod \hat{I}_j$ is a "shifted and reduced" version of f tagging along (with x as the only *free* variable).

Roots-tree data structure. Most importantly, we need to prove that $|\mathcal{L}|$, and the degree of the split ideals in $\mathcal{L}$, remains at most $\deg(f)$ at all times in the algorithm (while $f \bmod p^k$ may have exponentially many roots). To achieve this, we use a different way to look at the data structure $\mathcal{L}$ – in *tree* form RT where each generator h_i appearing in an $I \in \mathcal{L}$ appears as an edge of the tree; conversely, each tree node v denotes the intermediate split ideal corresponding to the path from the root (of the tree RT) to v.

The roots-tree RT has a useful parameter at every node– degree. Degree of a node measures the possible extensions to the next level, and it possesses the key property: it "distributes" to its children degrees. This helps us to simultaneously bound the width of RT and degree of split ideals, to be at most the degree $\deg(f)$ of the root node. Otherwise, since we compute with k-variate polynomials, a naive analysis of the tree-size (resp. degree of split ideals) would give a bound of $\deg(f)^k$, or a slightly better $\deg(f)2^k$ as in [6, pg.9]; which is exponential in the input size $\deg(f) \cdot k \log p$.

1.4 Proof overview

Proof idea of Theorem 1. Let $R := \mathbb{Z}/\langle p^k \rangle$; so $R/\langle p \rangle \cong \mathbb{F}_p$. Let $\mathcal{Z}_R(f)$ be the zeroset of $f \bmod p^k$.

The idea to count roots of $f \bmod p^k$ comes from the elementary fact: Any root $r \in R$ of $f \bmod p^k$ can be seen in a p-adic (or base-p) representation as $r =: r_0 + pr_1 + p^2r_2 + \ldots + p^{k-1}r_{k-1}$, for each $r_i \in \mathbb{F}_p$. Thus, we decompose our formal variable x into multi-variables $x_0, \ldots, x_{k-1}$ being related as, $x = x_0 + px_1 + p^2x_2 + \ldots + p^{k-1}x_{k-1}$.

Though, getting roots of $f(x_0) \bmod p$ deterministically is difficult, we can get the count on the number of roots of $f(x_0) \bmod p$ from the degree of a polynomial $h(x_0) \in \mathbb{F}_p[x_0]$, which is the gcd of f and *Frobenius* polynomial $x_0^p - x_0 \bmod p$. This way of implicitly representing a set of desired objects by a polynomial and using its properties (eg. degree) to get a count on the objects is widely termed as *polynomial method*.

This gives us a length-1 and degree-$\deg_{x_0}(h_0)$ split ideal $I_0 := \langle h_0(x_0) \rangle$. Since I_0 represents all roots of $f \bmod p$, we can again apply the polynomial method to incrementally build on ideal I_0 to get greater length split ideals representing roots of f with greater precision, say $\bmod p^{l+1}$.

To do this, we trivially lift I_0 to make it an ideal $\hat{I}_0$ in R. Solve $f(x_0 + px) \equiv p^\alpha g(x_0, x) \bmod \hat{I}_0$ for $\alpha \in \mathbb{N}$ and $g \not\equiv 0 \bmod p$. Reduce $g(x_0, x)$ over $\mathbb{F}_p$ again, and calculate the next set of candidates for x_1 implicitly in a polynomial $h_1 \in \mathbb{F}_p[x_0, x]$ defined as, $h_1 := \mathrm{GCD}(g(x_0, x) \bmod p, x^p - x) \bmod I_0$. Using the properties of split ideal (Lemma 11), multivariate-gcd modulo I_0 yields h_1 that "stores" all the candidates for x_1, for each root x_0 represented by I_0. So, we get a length 2 split ideal $I_1 := I_0 + \langle h_1(x_0, x_1) \rangle$.

In every iteration, we add a new variable, by solving equations like $f(x_0+px_1+p^2x_2+\ldots+p^lx_l+p^{l+1}x) \equiv p^\alpha g(\bar{x}_l, x)$ modulo a length $l+1$ triangular ideal $\hat{I}_l$, for $\alpha \in \mathbb{N}$ and $g \not\equiv 0 \bmod p$. This gives us the next candidate $h_{l+1}(\bar{x}_l, x) := \mathrm{GCD}(g(\bar{x}_l, x) \bmod p, x^p - x) \bmod I_l$; moving to a more precise split ideal. Sometimes we get that g and $x^p - x$ are coprime mod I_l, those cases indicate *dead-end* and we stop processing those branches. Finally we reach $\alpha = k$, which indicates *full precision*; and we get a maximal split ideal I_l which we add to the list $\mathcal{L}$.

Division by "zero". Some computations modulo a split ideal may not be possible. These cases arise only due to *zerodivisors*. In those cases, we will exploit the zerodivisor to split/factor the current split ideal into more split ideals of smaller degree. We can keep track of all these split ideals using a stack and keep performing the same computations iteratively. Since a split ideal has finite length, the process must terminate. The real challenge lies in proving a good bound.

Efficiency via roots-tree. Now, we need to show that the algorithm to construct $\mathcal{L}$ is efficient and that $|\mathcal{L}| \leq \deg(f)$ (in fact, sum of degrees of all maximal split ideals in $\mathcal{L}$ is at most $\deg(f)$). In a particular iteration, the algorithm just performs routine computations like– reduction modulo the current split ideal I, inversion, zerodivisor testing, gcd, exponentiation, and computing p-valuations or multiplicities; which are clearly bounded by $\mathrm{poly}(\deg(f), k \log p, \deg(I))$ (Sections C & D). It is harder to bound the number of iterations and $\deg(I)$.

To understand the number of iterations, we review the construction of $\mathcal{L}$ as the formation of a tree, which we call roots-tree RT. A node of RT corresponds to an intermediate split ideal I, where an edge at level i on the path from the root (of RT) to the node corresponds to the generator $h_i(\bar{x}_i)$ of I. Each time we update a split ideal I_{l-1} to $I_l := I_{l-1} + \langle h_l \rangle$ we add a child, to the node corresponding to I_{l-1}, hanging by a new edge labelled h_l. Similarly, splitting of an ideal at some generator $h_i(\bar{x}_i)$ into m ideals corresponds to creating m subtrees hanging by edges which are m copies of the edge labelled h_i. This way the roots-tree upper bounds the number of iterations; moreover, the maximal split ideals in $\mathcal{L}$ appear as leaves in RT.

Degree distribution in RT. Each node N of RT has an associated parameter, "degree of node" $[N]$ (Definition 15), which is defined in such a way that it distributes to degree of its children (i.e. $[N]$ is at least the sum of degrees of its child nodes). This is intended to measure the possible extensions x_l modulo the corresponding split ideal I_{l-1}, and is a suitable multiple of $\deg(I_{l-1})$. Applying degree's property inductively, we get that the degree of root node of RT, which is $\deg(f)$, distributes to the degree of the leaves and so the sum of degrees of all maximal split ideals in $\mathcal{L}$ is at most $\deg(f)$. The distributive property of $[N]$, corresponding to ideal I_{l-1}, comes from the fact: the degree of a child C corresponding to ideal $I_l = I_{l-1} + \langle h_l \rangle$ is bounded by the *multiplicity* of roots of $h_l(\bar{a}, x)$ times $\deg(I_{l-1})$, corresponding to some root $\bar{a}$ of I_{l-1}; and the overall sum of these multiplicities for every child of N is naturally bounded by the degree of N (Lemma 16).

The details are given in Section 3.

Proof idea of Theorem 2. The idea, and even the algebra, is the same as for Theorem 1. The definition of list $\mathcal{L}$ easily extends to implicitly store all the basic-irreducible factors of $f \bmod p^k$ of some degree b (a generalization over roots which corresponds to degree $b = 1$ basic-irreducible factors). This uses a strong property possessed by basic-irreducible factors. A basic-irreducible factor $g(x) \in (\mathbb{Z}/\langle p^k \rangle)[x]$ of $f \bmod p^k$, of degree b, completely splits over

the *Galois ring* $G(p^k, b) := \mathbb{Z}[y]/\langle p^k, \varphi(y)\rangle$, where $\varphi(y) \bmod p$ is an irreducible of degree b (Section A). Conversely, if we find a root of $f(x)$, in $G(p^k, b)$, then we find a degree-b basic-irreducible factor of $f \bmod p^k$.

By distinct degree factorization we can assume $f(x) \equiv (\varphi_1 \dots \varphi_m)^e + ph(x) \bmod p^k$, where each $\varphi_i(x) \bmod p$ is irreducible and degree-b. We construct $\mathcal{L}$ by applying the algorithm of Theorem 1, with one change: every time to update a length-l split ideal I_{l-1} to a length $l+1$ ideal $I_l := I_{l-1} + \langle h_l \rangle$, we compute h_l using the Frobenius polynomial $x^q - x \bmod p$, where $q := p^b$. Basically, for x, we focus on $\mathbb{F}_q$-roots instead of the erstwhile $\mathbb{F}_p$-roots.

We count the number of (distinct, monic, degree-b) basic-irreducible factors represented by each maximal split ideal $I(l, D) \in \mathcal{L}$ as: Dq^{k-l}/b. The details are given in Section 4.

1.5 Comparison with previous works

Broad outline. As already mentioned in Section 1.1, [2] was the first work to give an efficient algorithm to count, as well as find, all the roots of $f \bmod p^k$. The outline of [2] is as follows:

(1) Consider the roots of $f \bmod p^k$ in p-adic form as $r =: \Sigma_{i=0}^{k-1} r_i p^i$.

(2) Design an algorithm which builds up root r incrementally by finding candidates for the r_i's. The analysis of this gives rise to an exploration tree (which we call *roots-tree*).

(3) Bound the size of the roots-tree to show the efficiency of the algorithm.

At the most basic level, all the subsequent works, i.e. [6] and [19], as well as the present work, follow the above outline.

Previous randomized algorithms. The crux of the algorithm of [2], is that at every iteration it reduces finding the roots of $f(x) \bmod p^k$ to $f_a(x) \bmod p^{k-\alpha}$, where $f_a(x)$ is an integral polynomial defined as $f_a(x) := f(a + px)/p^\alpha$, for every root a of $f \bmod p$ and $p^\alpha || f(a + px)$. They used known randomized algorithms over $\mathbb{F}_p$ to get each such root a.[2]

The efficiency of their algorithm was based upon the observation that the degree of a node represented by f_a in the roots-tree is at most the multiplicity of the root a. This way [2] could show that, although exponential, the roots of $f \bmod p^k$ can be divided into at most d many easily representable clusters. The randomized root counting algorithm of [19] also follows the same multiplicity argument to show the efficiency of their algorithm ([19, Lem.3.6]). Unlike [2], even with randomization, [19] only do root-counting mod p^k.

Challenges in derandomization. It is challenging to extend the properties, of the randomized algorithms, to the deterministic (poly-time) context. The big questions that remained open were –

(1) Can we still cluster the roots of $f \bmod p^k$ deterministically? Note that efficient root finding algorithm over $\mathbb{F}_p$ is unavailable now in deterministic context.

(2) Can we get (may be implicitly) d clusters, deterministically?

(3) Can we generalize the multiplicity argument, first introduced in [2], in the deterministic context to show that size of the corresponding roots-tree is small?

(4) Can we extend the techniques to count basic-irreducible factors $f \bmod p^k$, deterministically?

[2] We refer to [13, Appdx.B] for a simple exposition of the algorithm of [2], after replacing φ by p there.

Deterministic Algorithms. Prior to our work, [6] was the best known deterministic algorithm (taking time exponential in k) for root-counting. The work of [6] was the first to give the idea of storing roots implicitly in ideals of triangular form, which we call split ideals. In that sense, the outline of both [6] and our work are the same – start with the split ideal of length one and keep growing it into more and more split ideals until we get all the maximal split ideals.

However, our algorithm to construct all the maximal split ideals, in particular *growing* and *splitting* intermediate split ideals, are quite different from [6]. [6] relies on special Grobner basis computation, and ideal decomposition algorithms, to grow and split ideals. For example, unlike [6]'s complexity bound of $\text{poly}(\deg(f), 2^k \log p)$ to compute $f(x_0 + px_1 + \ldots + p^l x_l + p^{l+1}x) =: p^\alpha g(\bar{x}_l, x) \bmod I_l$, we manage to optimize the time-complexity to $\text{poly}(\deg(f), k \log p)$ deterministically.

In lieu of Grobner basis, our algorithm relies on some simple key properties of split ideals (we systematically define and emphasize the key property of split ideals in Section 2). We also develop the algebra to perform computation (such as reduction, gcd, and testing for zero divisors) modulo a split ideal efficiently. Given these properties and subroutines, growing a split ideal $I_l = \langle h_0(x_0), \ldots, h_l(\bar{x}_l)\rangle$ is just a gcd computation modulo I_l (see the proof idea Section 1.4). Whenever we have a factorization $h_i =: h_{i,1}h_{i,2}$ of some generator, we split I_l efficiently into two split ideals $I_1 = \langle h_0, \ldots, h_{i-1}, h_{i,1}, \ldots, h_l\rangle$ and $I_2 = \langle h_0, \ldots, h_{i-1}, h_{i,2}, \ldots, h_l\rangle$ (a key property of split ideals).

Our simplified algorithm gives rise to a different roots-tree than [6], and helps us to observe some key properties of our roots-tree (Section 3.3). Using these properties, we could generalize the multiplicity argument of [2] (also improvement of [19, Lemma 3.6]) which in turn helped in bounding the number of clusters and tree size.

We feel that our idea is quite natural as Algorithm 1 easily extends to unramified extensions, as was the case with [2], and this in turn gives the count of all basic-irreducible factors as well. Moreover, our methods apply to the p-adic context (as they work for arbitrary k).

2 Preliminaries

Here we introduce our main tool - "split ideals". Proofs for this section have been moved to Section B. Basic introduction to Galois rings (i.e. non-prime characteristic analog of finite fields), Hensel lifting, randomized factoring over finite fields, etc. have been moved to Section A.

We will be given a univariate polynomial $f(x) \in \mathbb{Z}[x]$ of degree d and a prime power p^k (for a prime p and a positive integer $k \in \mathbb{N}$). Wlog, we assume that f is monic over $\mathbb{F}_p$.

A tuple of variables $(x_0, \ldots, x_l)$ will be denoted by $\bar{x}_l$. Often, an $(l+1)$-variate polynomial $a(x_0, x_1, \ldots, x_l)$ will be written as $a(\bar{x}_l)$, and the polynomial ring $\mathbb{F}_p[x_0, \ldots, x_l]$ as $\mathbb{F}_p[\bar{x}_l]$.

We denote the ring $\mathbb{Z}/\langle p^k\rangle$ by R (ring $R/\langle p\rangle$ is the same as field $\mathbb{F}_p$). An element $a \in R$ can be seen in its p-adic representation as $a = a_0 + pa_1 + \ldots + p^{k-1}a_{k-1}$, where $a_i \in \mathbb{F}_p$ for $i \in \{0, \ldots, k-1\}$.

$\mathcal{Z}_R(g) := \{r \in R \mid g(r) \equiv 0 \bmod p^k\}$ denotes the *zeroset* of a polynomial $g(x) \in R[x]$.

Zeroset of an ideal $I \subseteq \mathbb{F}_p[x_0, \ldots, x_l]$ is defined as the intersection of zeroset of all polynomials in I, $\mathcal{Z}_{\mathbb{F}_p}(I) := \{\bar{a} = (a_0, \ldots, a_l) \in (\mathbb{F}_p)^{l+1} \mid g(\bar{a}) \equiv 0 \bmod p, \forall g \in I\}$.

We will heavily use ideals of the form $I := \langle h_0(\bar{x}_0), h_1(\bar{x}_1), \ldots, h_l(\bar{x}_l)\rangle$ satisfying the condition – for any $i \in [l+1]$ and $\bar{a} \in \mathcal{Z}_{\mathbb{F}_p}(\langle h_0(\bar{x}_0), h_1(\bar{x}_1), \ldots, h_{i-1}(\bar{x}_{i-1})\rangle)$, polynomial $h_i(\bar{a}, x_i)$ splits completely into distinct linear factors. They are formally defined as:

▶ **Definition 4** (Split ideal). *We will call a polynomial* monic wrt x *if the leading-coefficient is one. Given $f(x) \in R[x]$, an ideal I, in $\mathbb{F}_p[\bar{x}_l]$, is called a* split ideal wrt f mod p^k *if,*

1) *I is a* triangular ideal of length $l+1$*, meaning: $I =: \langle h_0(\bar{x}_0), h_1(\bar{x}_1), \ldots, h_l(\bar{x}_l)\rangle$, for some $0 \le l \le k-1$; $h_i(\bar{x}_i) \in \mathbb{F}_p[\bar{x}_i]$ is monic wrt x_i, for all $i \in \{0, \ldots, l\}$,*
2) *$|\mathcal{Z}_{\mathbb{F}_p}(I)| = \prod_{i=0}^{l} \deg_{x_i}(h_i)$, and*
3) *$\forall (a_0, \ldots, a_l) \in \mathcal{Z}_{\mathbb{F}_p}(I)$, $f(a_0 + pa_1 + \ldots + p^l a_l) \equiv 0 \bmod p^{l+1}$.*

The length of I *is $l+1$ and its* degree *is* $\deg(I) := \prod_{i=0}^{l} \deg_{x_i}(h_i)$.

Split ideal I relates to possible roots of f mod p^k. Since f, p, k are fixed, we will call I a *split ideal*. The definition of a split ideal implies that its roots represent a set of "potential" roots of f, i.e. roots of f modulo some p^{l+1} for $0 \le l < k$. Restriction of a split ideal is also a split ideal.

▶ **Lemma 5** (Restriction of a split ideal). *Let $I_l := \langle h_0(\bar{x}_0), \ldots, h_l(\bar{x}_l)\rangle$ be a split ideal in $\mathbb{F}_p[x_0, \ldots, x_l]$, then ideal $I_j := \langle h_0(\bar{x}_0), \ldots, h_j(\bar{x}_j)\rangle$ is also a split ideal in $\mathbb{F}_p[x_0, \ldots, x_j]$, for all $0 \le j \le l$.*

Further, we show that a split ideal I can be decomposed in terms of its zeros.

▶ **Lemma 6** (Split ideal structure). *A split ideal $I \subseteq \mathbb{F}_p[x_0, \ldots, x_l]$ can be decomposed as $I = \bigcap_{\bar{a} \in \mathcal{Z}_{\mathbb{F}_p}(I)} I_{\bar{a}}$, where each $I_{\bar{a}} := \langle x_0 - a_0, \ldots, x_l - a_l\rangle$ corresponds to root $\bar{a} =: (a_0, \ldots, a_l) \in \mathcal{Z}_{\mathbb{F}_p}(I)$. By Chinese remainder theorem, $R/I = \bigoplus_{\bar{a} \in \mathcal{Z}_{\mathbb{F}_p}(I)} R/I_{\bar{a}}$.*

Let $I =: \langle h_0(\bar{x}_0), h_1(\bar{x}_1), \ldots, h_l(\bar{x}_l)\rangle$ be a split ideal. Suppose some h_i factors as $h_i(\bar{x}_i) = h_{i,1}(\bar{x}_i) \ldots h_{i,m}(\bar{x}_i)$. Define $I_j := \langle h_0(\bar{x}_0), \ldots, h_{i-1}(\bar{x}_{i-1}), h_{i,j}(\bar{x}_i), h_{i+1}(\bar{x}_{i+1}), \ldots, h_l(\bar{x}_l)\rangle$, for $j \in [m]$. The following corollary of Lemma 6 is evident because root-sets of I_j partition the root-set of I.

▶ **Corollary 7** (Splitting split ideals). *Let $I = \langle h_0(\bar{x}_0), \ldots, h_l(\bar{x}_l)\rangle$ be a split ideal of $\mathbb{F}_p[x_0, \ldots, x_l]$. Let some $h_i(\bar{x}_i)$ factor as $h_i(\bar{x}_i) = h_{i,1}(\bar{x}_i) \ldots h_{i,m}(\bar{x}_i)$.*

Then, $I = \bigcap_{j=1}^{m} I_j$, where each $I_j := \langle h_0(\bar{x}_0), \ldots, h_{i-1}(\bar{x}_{i-1}), h_{i,j}(\bar{x}_i), h_{i+1}(\bar{x}_{i+1}), \ldots, h_l(\bar{x}_l)\rangle$ is a split ideal.

We call a split ideal $I_l := \langle h_0, \ldots, h_l\rangle$ to be *maximal split ideal* if,

1) for any $\bar{a} = (a_0, \ldots, a_l) \in \mathcal{Z}_{\mathbb{F}_p}(I_l)$, $g(x) := f(a_0 + pa_1 + \ldots + p^l a_l + p^{l+1}x)$ vanishes identically mod p^k,
2) the restriction $I_{l-1} := \langle h_0, \ldots, h_{l-1}\rangle$ does not follow the previous condition.

▶ **Lemma 8** (Roots represented by a root of maximal split ideal). *Let I be a maximal split ideal of length $l+1$, then a zero $\bar{a} = (a_0, \ldots, a_l) \in \mathcal{Z}_{\mathbb{F}_p}(I)$ maps to exactly p^{k-l-1} zeros of f in $\mathcal{Z}_R(f)$. We will say that these p^{k-l-1} roots of f are* represented *by $\bar{a}$.*

3 Proof of Theorem 1

The algorithm to compute a compact data-structure which stores roots of f mod p^k will be described in Section 3.1. Algorithm's correctness will be proved in Section 3.2, which involves studying the algebraic structure underlying the algorithm. Its efficiency will be shown in Section 3.3, by devising an auxiliary structure called roots-tree and the important notion of "degree of a node".

3.1 Algorithm to implicitly partition the root-set of $f(x) \bmod p^k$

We describe our algorithm in this section. It takes a monic univariate polynomial $f(x) \in \mathbb{Z}[x]$ of degree d and a prime-power p^k as input (in binary), and outputs a list of at most d maximal split ideals whose roots partition the root-set of f modulo p^k.

A maximal split ideal $I_j =: \langle h_0(\bar{x}_0), \ldots, h_l(\bar{x}_l)\rangle$ has $|\mathcal{Z}_{\mathbb{F}_p}(I_j)| = \prod_{i=0}^{l} \deg_{x_i}(h_i)$ zeros, and each such zero "represents" p^{k-l-1} actual zeros of $f \bmod p^k$ (Lemma 8). Thus, this algorithm gives an exact count on the number of zeros of f in R.

Overview of Algorithm 1. Since any root of $f \bmod p^k$ is an extension of a root modulo p, the algorithm starts by initializing a stack S with the ideal $I := \langle h_0(x_0)\rangle$, where $h_0(x_0) := \gcd(x_0^p - x_0, f(x_0))$. This is a split ideal containing all the roots of $f \bmod p$. By a *lift* $\hat{I} \subset R[x_0]$ of I, we mean the ideal generated by the generator $\{h_0\}$ when viewed as a polynomial in $R[x_0]$ (i.e. char p^k).

At every intermediate iteration (Steps $4-21$), we *pop* a split ideal from the stack and *try* to increase the precision of its root-set (equivalently, lengthen the split ideal). This step mostly results in two cases: either we succeed and get a split ideal whose root-set has increased precision (Step 18) by a new placeholder x_{l+1}, or the split ideal factors into more split ideals increasing the size of the stack S (Steps $10, 14, 20$). We update the relevant "part of f" to $f_I(\bar{x}_l, x_{l+1} + px) \bmod \hat{J}$ (J is the new split ideal) that we carry around with each split ideal. This helps in efficiently increasing the precision of roots in the next iteration. Otherwise, computing $f\left(x_0 + px_1 + \cdots + p^l x_l + p^{l+1}x\right)/p^\alpha \bmod I$ is too expensive, in Step 6, due to the underlying degree-d $(l+1)$-variate monomials blowup.

If we reach a maximal split ideal (Step 7), it is moved to a list $\mathcal{L}$. Sometimes the split ideal cannot be extended and we get a *dead-end* (Step 16). The size of the stack decreases when we get a maximal split ideal or a dead-end. The algorithm terminates when stack becomes empty. List $\mathcal{L}$ contains maximal split ideals which partition, and cover, the root-set of f (implicitly). This becomes our output.

The main intuition behind our algorithm: If two roots of a split ideal (representing potential roots of f) give rise to different number of roots of f, the split ideal will get factored further. Though not at all apparent immediately, we will show that the algorithm takes only polynomial number of steps (Section 3.3).

We will use four subroutines to perform standard ring arithmetic modulo split ideals; they are described in the Appendices C & D.

1. Modify f (Steps 3, 18, 20) whenever pushing in the stack (Lemma 30 & 31).
2. Reduce$(a(\bar{x}_l), J_l)$ gives the reduced form of a mod triangular ideal J_l (over a Galois ring).
3. Test-Zero-Div$(a(\bar{x}_l), I_l)$ either reports that a is a not a zero-divisor modulo triangular ideal I_l or outputs a non-trivial factorization of one of the generators of I_l when true.
4. GCD$(a(\bar{x}_l, x), b(\bar{x}_l, x), I_l)$ either successfully computes a monic gcd, wrt x, of two multivariates modulo a triangular ideal I_l, or encounters a zerodivisor in intermediate computation (outputting $False$ and a non-trivial factorization of one of the generators of I_l).

Algorithm 1 Root-counting mod p^k.

Let $\mathcal{L} = \{\}$ be a list and $S = \{\}$ be a stack (both initially empty).
Let $\tilde{f}(x_0) := f(x_0) \bmod p$ for a monic univariate $\tilde{f} \in \mathbb{F}_p[x_0]$ of degree d.
[**Initializing the stack S**] Let $h_0(x_0) := \gcd(\tilde{f}(x_0), x_0^p - x_0)$, $I := \langle h_0 \rangle$, $\hat{I} \subseteq R[x_0]$ be a lift of I. Compute $f_I(x_0, x) := f(x_0 + px) \bmod \hat{I}$ using Lemma 30. Update $S \leftarrow push((\{h_0\}, f_I))$.
while S is not empty **do**
    $S_{top} \leftarrow pop(S)$. Let $S_{top} = (\{h_0(x_0), \ldots, h_l(x_0, \ldots, x_l)\}, f_I(\bar{x}_l, x))$ where $I = \langle h_0, \ldots, h_l \rangle \subseteq \mathbb{F}_p[\bar{x}_l]$ is a split ideal. Let $\hat{I} \subseteq R[x_0, \ldots, x_l]$ be a lift of I.
    [**Valuation computation**] Compute $\alpha \in \mathbb{N}$ and $g \in R[\bar{x}_l, x]$ such that $f_I \equiv p^\alpha g(\bar{x}_l, x) \bmod \hat{I}$ and $p \not| g \bmod \hat{I}$.
    [**Maximal split ideal found**] **if**$(\alpha \geq k)$ **then** update List $\mathcal{L} \leftarrow \mathcal{L} \cup \{I\}$. Go to Step 4.
    Let $\tilde{g} := g(\bar{x}_l, x) \bmod I$ be the polynomial in $\mathbb{F}_p[\bar{x}_l, x]$, and let $g_1(\bar{x}_l)$ be the leading coefficient of $\tilde{g}(\bar{x}_l, x)$ wrt x.
    if TEST-ZERO-DIV$(g_1(\bar{x}_l), I) = True$ **then**
        TEST-ZERO-DIV$(g_1(\bar{x}_l), I)$ returns a factorization $h_i(\bar{x}_i) =: h_{i,1}(\bar{x}_i)h_{i,2}(\bar{x}_i)\ldots h_{i,m}(\bar{x}_i) \bmod I_{i-1}$ of some generator $h_i(\bar{x}_i)$ of I. Go to Step 20.
    end if
    [**Filter out distinct virtual $\mathbb{F}_p$-roots by taking gcd with $x^p - x$**]
    Recompute $\tilde{g} := g(\bar{x}_l, x) \cdot g_1(\bar{x}_l)^{-1} \bmod I$ (Lemmas 29, 28). Compute x^p by repeatedly squaring and reducing modulo the triangular ideal $I + \langle \tilde{g} \rangle$ (Algorithm 2 and Lemma 28). This yields $\tilde{h}_{l+1}(\bar{x}_l, x) := x^p - x \bmod I$ in a reduced form.
    if GCD$(\tilde{g}, \tilde{h}_{l+1}, I) = False$ **then**
        The call GCD$(\tilde{g}, \tilde{h}_{l+1}, I)$ returns factorization $h_i(\bar{x}_i) = h_{i,1}(\bar{x}_i)h_{i,2}(\bar{x}_i)\ldots h_{i,m}(\bar{x}_i) \bmod I_{i-1}$ of a generator $h_i(\bar{x}_i)$ of I. Go to Step 20.
    else if $\tilde{g}$ and $\tilde{h}_{l+1}$ are coprime **then**
        [**Dead End**] The ideal I cannot grow more, go to Step 4.
    else
        [**Grow the split ideal** I] Here $\gcd_x(\tilde{g}, \tilde{h}_{l+1}) \bmod I$ is non-trivial, say $h_{l+1}(\bar{x}_l, x)$ (monic wrt x). Substitute x by x_{l+1} in $h_{l+1}(\bar{x}_l, x)$ and update $J \leftarrow I + \langle h_{l+1}(\bar{x}_{l+1}) \rangle$. Let $\hat{J} \subseteq R[x_0, \ldots, x_{l+1}]$ be a lift of J. Substitute x by $x_{l+1} + px$ in $f_I(\bar{x}_l, x)$, and compute $f_J(\bar{x}_{l+1}, x) := f_I(\bar{x}_l, x_{l+1} + px) \bmod \hat{J}$ using Lemma 30. Update $S \leftarrow push((\{h_0, \ldots, h_{l+1}\}, f_J))$, and go to Step 4.
    end if
    [**Factoring split ideals**] We have a factorization $h_i(\bar{x}_i) = h_{i,1}(\bar{x}_i)h_{i,2}(\bar{x}_i)\ldots h_{i,m}(\bar{x}_i) \bmod I_{i-1}$ of a generator h_i of I. Push S_{top} back in stack S. For every entry $(U, f_{\langle U \rangle}) \in S$, where $h_i(\bar{x}_i)$ appears in U, find m (smaller) split ideals U_j (using Corollary 7); using Lemma 31 compute $f_{\langle U_j \rangle}$ and push $(U_j, f_{\langle U_j \rangle})$ in S, for $j \in [m]$.
end while
Return $\mathcal{L}$ (the list of maximal split ideals partitioning the root-set $\mathcal{Z}_R(f)$).

3.2 Correctness of Algorithm 1

Our main goal is to prove the following result about partitioning of root-set.

▶ **Theorem 9** (Algo 1 partitions $\mathcal{Z}_R(f)$)**.** *Algorithm 1 yields the structure of the root-set $\mathcal{Z}_R(f)$ through a list data structure $\mathcal{L}$ (a collection of maximal split ideals $I_1, \ldots, I_n$) which partitions the zeroset $\mathcal{Z}_R(f) =: \bigsqcup_{j\in[n]} S_j$, where S_j is the set of roots of $f \bmod p^k$ represented by $\mathcal{Z}_{\mathbb{F}_p}(I_j)$.*

Later, we will show a surprising property: $n \leq d$ (Section 3.3).

Proof of Theorem 9. Lemma 12 states that at any point, Stack S only contains split ideals which have disjoint root sets. Lemma 13 assures that Algorithm 1 terminates on every input. So from Lemmas 12, 13 and the definition of maximal split ideal, it is clear that Algorithm 1 returns a list $\mathcal{L}$ containing maximal split ideals $I_1, \ldots, I_n$, for $n \in \mathbb{N}$. Further, we show:
1) The root-set of I_j $(1 \leq j \leq n)$ yields a subset S_j of $\mathcal{Z}_R(f)$, and they are pairwise disjoint.
2) Given a root $r \in \mathcal{Z}_R(f)$, there exists j such that r is represented by a root in $\mathcal{Z}_{\mathbb{F}_p}(I_j)$.

For the first part, root-sets for different maximal split ideals I_j are pairwise disjoint because of Lemma 12. Each of these root-set yields a subset of the zeroset of $f \bmod p^k$ (follows from the definition of maximal split ideal).

For the second part, let $r =: \sum_{i=0}^{k-1} r_i p^i$ be a root in $\mathcal{Z}_R(f)$. Stack S was initialized by the split ideal $\langle h_0 := \gcd(f(x_0) \bmod p, x_0^p - x_0)\rangle$; so $r_0 \in \mathcal{Z}_{\mathbb{F}_p}(I_0)$, as $f(r_0) \equiv f(r) \equiv 0 \bmod p$.

Assume that I_0 is not a maximal split ideal (otherwise we are done). Applying Lemma 14, there must exist an I_1 whose root-set contains (r_0, r_1). Repeated applications of Lemma 14 show that we will keep getting split ideals of larger lengths, partially representing r; finally, reaching a maximal split ideal (say I_j) fully representing r.

We showed that each root r of $f \bmod p^k$ is represented by a *unique* maximal split ideal I, given by Algorithm 1, and they collectively represent exactly the roots of f modulo p^k. Hence, root-sets of ideals in $\mathcal{L}$ partition the zeroset $\mathcal{Z}_R(f)$. ◀

Now, let us see the properties of our algorithm which go in proving Theorem 9. Given a polynomial $g(\bar{x}_l) \in \mathbb{F}_p[\bar{x}_l]$ and an element $\bar{a} \in \mathbb{F}_p^l$, consider the *projection* $g_{\bar{a}}(x_l) := g(\bar{a}, x_l)$. Using Chinese remainder theorem (Lemma 6) we easily get the following degree condition. (Here, lc_x refers to the leading coefficient wrt variable x.)

▷ Claim 10. Let I be a split ideal of $\mathbb{F}_p[\bar{x}_{l-1}]$ and $g \in \mathbb{F}_p[\bar{x}_l]$. Then, $\mathrm{lc}_{x_l}(g)$ is unit mod I iff $\forall \bar{a} \in \mathcal{Z}_{\mathbb{F}_p}(I),\ \deg(g_{\bar{a}}(x_l)) = \deg_{x_l}(g(\bar{x}_l) \bmod I)$.

Chinese remaindering also gives us a gcd property under projections.

▶ **Lemma 11.** *Let $w(\bar{x}_l), z(\bar{x}_l) \in \mathbb{F}_p[\bar{x}_l]$ and $I_{l-1} \subseteq \mathbb{F}_p[\bar{x}_{l-1}]$ be a split ideal. Suppose Algorithm 4 succeeds in computing gcd of w and z mod I_{l-1}: define $h(\bar{x}_l) := \mathrm{GCD}(w(\bar{x}_l), z(\bar{x}_l), I_{l-1})$. Then, for all $\bar{a} \in \mathcal{Z}_{\mathbb{F}_p}(I_{l-1})$: $h_{\bar{a}}(x_l)$ equals $\gcd(w_{\bar{a}}(x_l), z_{\bar{a}}(x_l))$ up to a unit multiple (in $\mathbb{F}_p^*$).*

Proof. Lemma 33 proves, $h(\bar{x}_l)$ is a monic polynomial mod I_{l-1}, s.t., $h|w$ and $h|z$ (mod I_{l-1}). Fix $\bar{a} \in \mathcal{Z}_{\mathbb{F}_p}(I_{l-1})$. Since $h_{\bar{a}}(x_l) \not\equiv 0 \bmod p$ ($\because h$ is monic), restricting $\bar{x}_{l-1}$ to $\bar{a}$ gives $h_{\bar{a}}|w_{\bar{a}}$ and $h_{\bar{a}}|z_{\bar{a}}$, showing $h_{\bar{a}}|\gcd(w_{\bar{a}}, z_{\bar{a}})$, in $\mathbb{F}_p[x_l]$.

Lemma 33 also shows that there exists $u, v \in (\mathbb{F}_p[\bar{x}_{l-1}]/I_{l-1})[x_l]$, such that, $h = uw + vz$. Restricting first l co-ordinates to $\bar{a}$, we get $h_{\bar{a}} = u_{\bar{a}} w_{\bar{a}} + v_{\bar{a}} z_{\bar{a}}$. This equation implies $\gcd(w_{\bar{a}}, z_{\bar{a}})|h_{\bar{a}}$. Thus, we get an equality up to a unit multiple. ◀

Let $I \subseteq \mathbb{F}_p[\bar{x}_i], J \subseteq \mathbb{F}_p[\bar{x}_j]$ be two split ideals (say $i \leq j$). I and J are called *prefix-free* iff $\nexists\ \bar{a} = (a_0, a_1, \ldots, a_i) \in \mathcal{Z}_{\mathbb{F}_p}(I),\ \bar{b} = (b_0, b_1, \ldots, b_j) \in \mathcal{Z}_{\mathbb{F}_p}(J) : a_k = b_k\ \forall k \leq i$.

(Note that it may still happen that $(a_0, \ldots, a_{i-1}) = (b_0, \ldots, b_{i-1})$ above.)

Our next lemma shows an invariant about Algorithm 1.

▶ **Lemma 12** (Stack contents). *Stack S in Algorithm 1 satisfies following conditions at every point:*

1) *$l < k$ and in Step 6, $\alpha > l$.*
2) *All ideals in S are split ideals.*
3) *Any two ideals in S are prefix-free.*

Proof. We first prove the invariant 1. Step 6 defines g via f_I as, $f_I =: p^\alpha g(\bar{x}_l, x) \bmod \hat{I}$. Looking at the f_I analogues pushed in Steps 3, 18, 20, one easily deduces the invariants:
$f\left(\sum_{0\leq i\leq l} x_i p^i + x p^{l+1}\right) \equiv f_I(\bar{x}_l, x) \bmod \hat{I}$, and
$f\left(\sum_{0\leq i\leq l} x_i p^i\right) \equiv 0 \bmod \hat{I} + \langle p^{l+1}\rangle$.

Thus, $f\left(\sum_{0\leq i\leq l} x_i p^i\right) \equiv p^\alpha g(\bar{x}_l, x) \equiv 0 \bmod \hat{I} + \langle p^{l+1}\rangle$. Since, $p \nmid g \bmod \hat{I}$, we deduce $\alpha > l$. Moreover, by Step 7 we know that $l < k$ throughout the algorithm.

There are three ways in which a new ideal is added to stack S. We show below that the invariant is maintained in all three cases.

(Step 3) S is initialized with the ideal $I = \langle h_0(x_0)\rangle \subseteq \mathbb{F}_p[x_0]$. The triangular ideal I is a split ideal, because $|\mathcal{Z}_{\mathbb{F}_p}(I)| = \deg_{x_0}(h_0)$ and its root are all the distinct roots of $f(x_0) \bmod p$.

(Step 20) Ideal I_l is popped from S, and some generator h_i of I_l splits. In this case, we update S with the corresponding factors of any $(U, f_{\langle U\rangle}) \in S$, wherever currently U has h_i. Corollary 7 shows that the factors of U are split ideals themselves, and their root-sets partition that of U. Thus, these root-sets are prefix-free among themselves. Moreover, they are prefix-free with any other ideal J appearing in S, because U was prefix-free with J.

(Step 18) Ideal I_l is popped, it grows to I_{l+1} by including $h_{l+1}(\bar{x}_l, x) = \gcd_x(\tilde{g}(\bar{x}_l, x), x^p - x) \bmod I_l$ ($\tilde{g}$ is defined in Step 8). First (resp. third) condition for I_{l+1} being a split ideal follows from the definition of $\tilde{g}$ (resp. h_{l+1}).

For the second condition for I_{l+1} being a split ideal, fix a particular root $\bar{a} \in \mathcal{Z}_{\mathbb{F}_p}(I_l)$. Using Lemma 11, the projection $h_{l+1,\bar{a}}(x)$ equals $\gcd(\tilde{g}_{\bar{a}}(x), x^p - x)$ (up to a unit multiple). By Lemma 33, h_{l+1} is monic mod I_l; giving $\deg(h_{l+1,\bar{a}}) = \deg_{x_{l+1}}(h_{l+1})$. Since $h_{l+1} | x^p - x$, there are exactly $\deg_x(h_{l+1})$-many $a_{l+1} \in \mathbb{F}_p$, such that $h_{l+1,\bar{a}}(a_{l+1}) \equiv 0 \bmod p$. So, every root $\bar{a} \in \mathcal{Z}_{\mathbb{F}_p}(I_l)$ can be extended to $\deg_x(h_{l+1})$-many roots; giving $|\mathcal{Z}_{\mathbb{F}_p}(I_{l+1})| = \deg_x(h_{l+1}) \cdot \prod_{i=0}^{l} \deg_{x_i}(h_i)$. This makes I_{l+1} a split ideal.

I_{l+1} remains prefix-free with any other ideal J of S, because roots of I_{l+1} are extension of roots of I_l (recall: I_l was prefix-free with J and it was popped out of S).

This proves all the invariants for the stack S. ◀

Using the invariant, we prove that Algorithm 1 terminates on any input.

▶ **Lemma 13.** *Algorithm 1 finishes in finite number of steps for any $f \in \mathbb{Z}[x]$ and a prime power p^k.*

Proof. We show that the number of iterations in Algorithm 1 are finite. Assume that all the ideals which result in a dead-end are moved to a list D; say C is the disjoint union of all ideals in S, $\mathcal{L}$ and D. Whenever a split ideal I from S is moved to $\mathcal{L}$ or D, the underlying roots (of I) stop extending to the next precision. Togetherwith Lemma 12, we deduce that in fact all the ideals in C are prefix-free. Now by Step 18, and the rate of growth of split ideals up to length $l + 1 \leq k$, we get a lazy estimate of $|C| \leq \min(d^k, p^k)$.

Let $\mathrm{len}(I)$ denote the length of an ideal I, it is bounded by k. Notice that factoring/growing an ideal increases $\sum_{I\in C}\mathrm{len}(I)$; and getting a maximal split ideal/ dead-end increases $|\mathcal{L}|+|D|$. Thus, every iteration of the algorithm strictly increases the quantity $(\sum_{I\in C}\mathrm{len}(I))+|\mathcal{L}|+|D|$. By the estimate on $|C|$, all the terms in this quantity are bounded; thus, the number of iterations are finite. ◀

The following lemma shows: if we see a restriction of $r \in \mathcal{Z}_R(f)$ (say, up to length $l+1$) at some point in Algorithm 1, we will again see its restriction of length $l+2$ at a later point in the algorithm.

▶ **Lemma 14** (Getting roots with more precision). *Assume that at some time (say t), Algorithm 1 pops an ideal I of length $l+1$, that is not yet a maximal split ideal. Let $\bar{a} = (a_0, \ldots, a_l) \in \mathcal{Z}_{\mathbb{F}_p}(I)$ partially represent a "root" $r =: \sum_{0\le i\le l+1} a_i p^i$ such that $f(r) \equiv 0 \mod p^{l'}$, but $f\left(\sum_{0\le i\le l} a_i p^i + x p^{l+1}\right) \not\equiv 0 \mod p^{l'}$, for some $l+2 \le l' \le k$. Then, there exists a time $t' > t$, when stack S will pop an ideal J of length $l+2$, such that, $(\bar{a}, a_{l+1}) \in \mathcal{Z}_{\mathbb{F}_p}(J)$.*

Proof. We again consider three possible situations.

(Step 18) Ideal I grows to another split ideal, say J. Notice, J is obtained by adding $h_{l+1} := \mathrm{GCD}(g(\bar{x}_l, x), x^p - x) \bmod I$ to I (setting $x \mapsto x_{l+1}$).

Step 6 defines g via f_I as, $f_I =: p^\alpha g(\bar{x}_l, x) \bmod \hat{I}$. Looking at the f_I analogues pushed in Steps 3, 18, 20, one can deduce the invariant: $f\left(\sum_{0\le i\le l} x_i p^i + x p^{l+1}\right) \equiv f_I(\bar{x}_l, x) \bmod \hat{I}$.

Now, let us project to (suitable integral lifts of) $\bar{a}$ and consider $f\left(\sum_{0\le i\le l} a_i p^i + x p^{l+1}\right) \equiv f_I(\bar{a}, x) \equiv p^\alpha g(\bar{a}, x) \bmod \hat{I}$. By Step 9, and Claim 10, we are assured that $g(\bar{a}, x)$, $g(\bar{x}_l, x) \bmod I$ are equi-degree (wrt x). Thus, by non-maximality hypothesis we have $\alpha < l'$. Hypothesis tells us that $f\left(\sum_{0\le i\le l+1} a_i p^i\right) \equiv 0 \mod p^{l'}$. So, by the previous paragraph, $p^\alpha g(\bar{a}, a_{l+1}) \equiv 0 \mod p^{l'}$. Whence, $g(\bar{a}, a_{l+1}) \equiv 0 \bmod p$. Clearly, $a_{l+1}^p - a_{l+1} \equiv 0 \bmod p$. Thus, $h_{l+1}(\bar{a}, a_{l+1}) \equiv 0 \bmod p$. So $(\bar{a}, a_{l+1})$ is a root of J.

(Step 16) Proof of the previous case shows that $h_{l+1}(\bar{a}, x)$ has degree at least 1, so I could not result in a *dead-end*.

(Step 20) Ideal I factors into (smaller) split ideals. In this case, $\bar{a}$ will be included in exactly one of those ideals (by Corollary 7). This ideal will be handled later in the algorithm and will give an ideal J with $(\bar{a}, a_{l+1})$ as root. ◀

3.3 Time complexity of Algorithm 1 – introducing roots-tree RT

We know that Algorithm 1 takes finite amount of time and terminates (Lemma 13). To show that it is efficient, note that the time complexity of the algorithm can be divided into two parts.

1) Number of iterations taken by Algorithm 1, which is clearly bounded by the number of updates on Stack S in the algorithm.
2) Time taken by the various algebraic operations in one iteration of the algorithm: reduction by a triangular ideal, valuation computation modulo a split ideal, testing if some polynomial is a zerodivisor modulo a split ideal, performing repeated squaring modulo a triangular ideal and computing gcd of two multi-variates modulo a split ideal.

For the purpose of bounding iterations, we define a "virtual" tree, called *roots-tree (RT)*, which essentially keeps track of the updates on Stack S. We will map a node $N = (I, f_I)$ in roots-tree to the element (I, f_I) in stack S. Each *push* will create a new node in RT. The nodes are *never* deleted from RT.

Construction of roots-tree (RT). Denote the root of RT by $N_{\langle 0\rangle} := (\langle 0\rangle, f_{\langle 0\rangle} := f(x))$. Add a child node N_{I_0} to the root corresponding to the initialization of Stack S by (I_0, f_{I_0}), where $I_0 := \langle h_0(x_0)\rangle$ (label the edge h_0 in RT).

If, at some time t, the algorithm pops $(I_{l-1}, f_{I_{l-1}})$ from S then the *current node* in RT will be the leaf node $N_{I_{l-1}} = (I_{l-1}, f_{I_{l-1}})$. We map the updates on stack S to RT as follows:

(Step 18) If ideal I_{l-1} *grows* to $I_l := I_{l-1} + \langle h_l\rangle$ and (I_l, f_{I_l}) is pushed in S, then create a child of $N_{I_{l-1}}$ in RT using an edge labelled h_l (label the node $N_{I_l} := (I_l, f_{I_l})$).

(Steps 7, 16) If the algorithm reached *dead-end* (no update in stack S or list $\mathcal{L}$), then add a child labelled $\mathcal{D}$ to node $N_{I_{l-1}}$. It indicates a dead-end at the current branch. Analogously, if the algorithm finds a *maximal split ideal*, we add a child labelled $\mathcal{M}$ to Node $N_{I_{l-1}}$ (indicating I_{l-1} is a maximal split ideal).

(Step 20) Suppose, processing of length-l split ideal I_{l-1} results in factoring each ideal U in S, containing h_i, to m split ideals. We describe the *duplication process* for a particular U (repeat it for each split ideal containing h_i).

Let U_{i-1} be the length-i restriction of U. First, we move to the ancestor node $N_{U_{i-1}} := (U_{i-1}, f_{U_{i-1}})$ of N_U. Make m copies of the sub-tree at Node $N_{U_{i-1}}$, each of them attached to $N_{U_{i-1}}$ by edges labelled with $h_{i,1}, \ldots, h_{i,m}$ respectively. The copy of each old node $N = (V, f_V)$, in sub-tree corresponding to $h_{i,j}$, will be relabelled with (V_j, f_{V_j}) corresponding to the factor split ideal V_j of V and the newly computed f_{V_j}.

This step does not increase the height of the tree, though it increases the size.

For the rest of this section, RT denotes the final roots-tree created at the end of the above process.

Properties of RT. We state some easy properties of RT, which will help us in analyzing the time complexity.

1) By construction, size of the roots-tree increases at every iteration. We never delete a node or an edge (though relabelling might be done). So, the size of RT bounds the number of iterations taken by Algorithm 1.
2) Consider a node $N_I =: (I, f_I)$ in RT. Here $f_I(\bar{x}_l, x) \in R[\bar{x}_l, x]$, and let $g_I \in R[\bar{x}_l, x]$ be defined as in Algorithm 1, $g_I := f_I(\bar{x}_l, x)/p^\alpha \bmod \hat{I}$, where $p^\alpha \mid\mid f_I \mod \hat{I}$, and $\hat{I}$ is a lift of I over R. Then, $g_I \bmod I$ is a nonzero polynomial over $\mathbb{F}_p$.
3) For each node $N_I =: (I, f_I(\bar{x}_l, x))$ and its child $N_J =: (J, f_J(\bar{x}_{l+1}, x))$, we have the relation, $f_J = f_I(\bar{x}_l, x_{l+1} + px) \bmod \hat{J}$.

Bounding $|RT|$. To bound the size of RT, we define a parameter for a node N of RT, called the *degree* of the node N and denoted by $[N]$.

▶ **Definition 15** (Degree of a node in RT). *Degree of leaves $\mathcal{D}$ resp. $\mathcal{M}$ is defined to be* 1.

Let $N_I =: (I, f_I)$ be a node corresponding to a split ideal $I \subseteq \mathbb{F}_p[\bar{x}_l]$, where $f_I(\bar{x}_l, x)$ belongs to $R[\bar{x}_l, x]$. Let $p^\alpha \mid\mid f_I \bmod \hat{I}$ and $g_I(\bar{x}_l, x) := f_I/p^\alpha \bmod \hat{I}$. Except, $g_I := 0$ if $\alpha \geq k$.

Then, the degree of N is defined as, $[N] := \max(1, \deg_x(g_I \bmod I) \times \deg(I))$.

We remark that for the root node, $N_{\langle 0\rangle} = (\langle 0\rangle, f_{\langle 0\rangle} := f(x))$ we will set $\deg(\langle 0\rangle) := 1$. *Then, it is clear by the general definition of degree that $[N_{\langle 0\rangle}] = d$ $(= \deg(f) \geq 1)$.*

We show that the degree of a parent node bounds the sum of the degree of its children.

▶ **Lemma 16** (Degree distributes in RT). *Let N be a node in roots-tree RT and $des(N)$ denote the set of all children of N. Then, $[N] \geq \sum_{C \in des(N)} [C]$.*

So, the sum of the degrees of all nodes, at any level l, is at least the sum of the degrees of all nodes at level $l + 1$.

Proof. Let $N = (I, f_I)$, where $I = \langle h_0, \ldots, h_l \rangle$ and $f_I(\bar{x}_l, x) \in R[\bar{x}_l, x]$. Define $\tilde{g}_I \in \mathbb{F}_p[\bar{x}_l, x]$ as $\tilde{g}_I := g_I(\bar{x}_l, x) \bmod I$. Assume $\alpha < k$, otherwise we are done. So, $g_I \bmod I$ is nontrivial wrt x; by Step 9 (failure) and Claim 10, we get,

$$\forall \bar{a} \in \mathcal{Z}_{\mathbb{F}_p}(I) : \ \deg_x(\tilde{g}_I \bmod I) \ = \ \deg_x(\tilde{g}_I(\bar{a}, x)) \ . \tag{1}$$

Recall $h_{l+1}(\bar{x}_l, x) := \gcd(\tilde{g}_I(\bar{x}_l, x), x^p - x)$.Let C be a child node of N in RT such that $C =: (J_C, f_{J_C})$, where $J_C =: I + \langle h_{l,C}(\bar{x}_{l+1}) \rangle$ and $f_{J_C}(\bar{x}_{l+1}, x) := f_I(\bar{x}_l, x_{l+1} + px) \bmod \hat{J}_C$. This gives us the factorization $h_{l+1}(\bar{x}_l, x) \ = \ \prod_{C \in \mathrm{des}(N)} h_{l,C}(\bar{x}_l, x) \bmod I$ (Step 20, and "duplication step" when we constructed RT). Again,

$$\forall \bar{b} \in \mathcal{Z}_{\mathbb{F}_p}(J_C) : \ \deg_x(\tilde{g}_{J_C} \bmod J_C) \ = \ \deg_x(\tilde{g}_{J_C}(\bar{b}, x)) \ . \tag{2}$$

If $g_{J_C} =: f_{J_C}/p^{v'} \bmod \hat{J}_C$ for some $v' \in \mathbb{N}$, by property 3 of RT, we have $g_{J_C} = f_I(\bar{x}_l, x_{l+1} + px)/p^{v'} \bmod \hat{J}_C$.

By definition, $[N] = \deg(I) \cdot \deg_x(\tilde{g}_I)$ and $[C] = \deg(J_C) \cdot \deg_x(\tilde{g}_{J_C})$. Since $\deg(J_C) = \deg(I) \cdot \deg_x(h_{l,C}(\bar{x}_l, x))$, the lemma statement is equivalent to showing,

$$\deg_x(\tilde{g}_I) \ \geq \ \sum_{C \in \mathrm{des}(N)} \deg_x(h_{l,C}(\bar{x}_l, x)) \cdot \deg_x(\tilde{g}_{J_C}) \ . \tag{3}$$

Continuing with the notation of a particular child C, fix an $\bar{a} \in \mathcal{Z}_{\mathbb{F}_p}(I)$. Since J_C is a split ideal, $h_{l,C}(\bar{a}, x)$ (of degree d'_C) can be written as $\prod_{i=1}^{d'_C}(x - c_i)$, where each $c_i \in \mathbb{F}_p$ and are distinct. Then, each c_i is also a root of $\tilde{g}_I(\bar{a}, x)$, say with multiplicity $m_i \in \mathbb{N}$. So, there exists $G(x) \in \mathbb{F}_p[x]$ (coprime to $x - c_i$), such that, $\tilde{g}_I(\bar{a}, x) \ \equiv \ (x - c_i)^{m_i} \cdot G(x) \bmod p$. Lifting this equation mod p^k, there exists $G_1(x) \in R[x]$, of degree less than m_i, and a unique lift $G_2(x) \in R[x]$ of $G(x)$ (Hensel lemma (21)) : $g_I(\bar{a}, x) \ \equiv \ ((x - c_i)^{m_i} + pG_1(x)) \cdot G_2(x) \bmod p^k$. Substituting $x \to c_i + px$, we get, $g_I(\bar{a}, c_i + px) \ \equiv \ ((px)^{m_i} + pG_1(c_i + px)) \cdot G_2(c_i + px) \bmod p^k$.

Let $\bar{b}_i = (\bar{a}, c_i) \in \mathcal{Z}_{\mathbb{F}_p}(J_C)$. We know that $\tilde{g}_{J_C}(\bar{b}_i, x) = f_I(\bar{a}, c_i + px)/p^{v'} \bmod p$ is nontrivial. This implies that, $((px)^{m_i} + pG_1(c_i + px))/p^{v'} \bmod p$ is a nonzero polynomial of degree at most m_i ($\because p \nmid G_2(c_i)$).

Since $G_2(c_i + px) \not\equiv 0 \bmod p$ is a unit, $\deg_x(\tilde{g}_{J_C}(\bar{b}_i, x)) = \deg_x(\tilde{g}_{J_C}) \leq m_i$ (Eqn. 2). Summing up over all the roots c_i of $\tilde{g}_I(\bar{a}, x)$,

$$\sum_{i=1}^{d'_C} \deg_x(\tilde{g}_{J_C}(\bar{b}_i, x)) \ = \ d'_C \cdot \deg_x(\tilde{g}_{J_C}) \ \leq \ \sum_{i=1}^{d'_C} m_i \ =: d_C(g_I) \ .$$

Summing over all children $C \in \mathrm{des}(N)$ (using Eqn. 1, factorization of h_{l+1} & distinctness of $\mathbb{F}_p$-roots), we deduce,

$$\sum_{C \in \mathrm{des}(N)} \deg_x(h_{l,C}) \deg_x(\tilde{g}_{J_C}) \ \leq \ \sum_C d_C(g_I) \ \leq \ \deg_x(\tilde{g}_I(\bar{a}, x)) \ = \ \deg_x(\tilde{g}_I) \ .$$

This proves Eqn. 3, and hence the lemma. ◀

Define the degree of list $\mathcal{L}$ as, $\deg(\mathcal{L}) := \Sigma_{I \in \mathcal{L}} \deg(I)$.

▶ **Lemma 17** (Bounding $|RT|$, $\deg(I)$, $\deg(\mathcal{L})$, $|\mathcal{L}|$)**.** *Let RT be the roots-tree constructed from the execution of Algorithm 1. The number of leaves of RT, resp. $\deg(\mathcal{L})$, is at most $d = \deg(f(x))$. Also, the size $|RT|$ of the roots-tree (hence, the number of iterations by Algorithm 1) is bounded by dk.*

Proof. Applying Lemma 16 inductively, sum of the degrees of nodes at any level is bounded by the degree d of the root node. In particular,

1) We can extend every leaf to bring it to the last level (create a chain of nodes of same degree) without changing the degree distribution property. So, $\deg(\mathcal{L}) = \Sigma_{I\in\mathcal{L}} \deg(I) \leq d$. Since the number of leaves is $\geq |\mathcal{L}|$, we get $|\mathcal{L}| \leq d$.
2) For any split ideal I in stack S, $\deg I \leq d$.
3) Since the depth of the roots-tree is at most k, $|RT| \leq kd$. ◀

▶ **Lemma 18** (Computation cost at a node). *Computation cost at each node of RT (time taken by Algorithm 1 in every iteration of the while loop) is bounded by poly$(d, k \log p)$.*

Proof. During an iteration, the major computations performed by the algorithm are – testing for zerodivisors (Step 9), computing modular gcd (Step 13), computing reduced f_I (Steps 3, 18), performing reduction for repeated squaring (Step 12), and factoring ideals (Step 20).

These operations are described by Lemmas 28, 29, 30, 32 and 33. All of them take time poly$(d, k \log p, \deg(I))$, where I is the concerned triangular ideal.

For any split ideal I (or its lift $\hat{I}$), we know that $\deg(I) \leq d$ (Lemma 17). So, Steps 3, 9, 13, 18, 20 take time poly$(d, k \log p)$. Step 12 to compute repeated squaring modulo $I + \langle \tilde{g} \rangle$ takes time poly$(\deg_x(\tilde{g}), \deg(I), k \log p)$ (using Lemma 28). Since I is a split ideal with $\deg(I) \leq d$, and degree of $\tilde{g}$ is at most d, so Step 12 also takes poly$(d, k \log p)$ time.

Hence the computation cost at each node is poly$(d, k \log p)$. ◀

Proof of Theorem 1. The definition of roots-tree shows that the number of leaves upper bound the number of all maximal split ideals in $\mathcal{L}$. Lemmas 17 and 18 show that the time complexity of Algorithm 1 is bounded by poly$(d, k \log p)$ (by bounding both number of iterations and the cost of computation at each iteration). Using Lemma 8 on the output of Algorithm 1, we get the exact count on the number of roots of $f \bmod p^k$ in time poly$(d, k \log p)$. ◀

4 Proof of Theorem 2

A polynomial f can be factored mod p^k if it has two basic-irreducible factors of different degree (using distinct degree factorization [33] and Hensel Lemma 21).

If two basic-irreducible factors appear with different exponents/multiplicities, then again f can be factored (using formal derivatives [33] and Hensel Lemma 21).

So, for factoring $f \bmod p^k$, we can assume $f \equiv (\varphi_1 \dots \varphi_t)^e + ph \bmod p^k$, where every $\varphi_i \in (\mathbb{Z}/\langle p^k \rangle)[x]$ is a basic-irreducible polynomial of a fixed degree b. Also, $d := \deg(f) = bte$. Let us fix this assumption for this section, unless stated explicitly.

A basic-irreducible factor of $f \bmod p^k$ has the form $\varphi_i + pw_i(x) \bmod p^k$, for $i \in [t]$ (Lemma 22).

If $b = 1$, counting basic-irreducible factors of f is equivalent to counting roots of f.

When $b > 1$, we prove a simple generalization of this idea; it is enough to count all the roots of f in the ring extension $\mathbb{Z}[y]/\langle p^k, \varphi(y) \rangle$, where $\varphi(y)$ is an irreducible mod p of degree-b. These rings are called *Galois rings*, we denote them by $G(p^k, b)$ (unique, for fixed k and b, up to isomorphism).

4.1 Reduction to root-counting in $G(p^k, b)$

By Lemma 22, any basic-irreducible factor of $f \bmod p^k$ is a factor of a unique $({\varphi_i}^e + pw_i(x))$; and φ_i are coprime mod p. So in this subsection, for simplicity of exposition, we will assume that $f(x)$ equals $\varphi^e \bmod p$ (φ is a monic degree-b irreducible mod p).

Define $G := G(p^k, b)$. Let $y_0, y_1, \ldots, y_{b-1}$ be the roots of $\varphi(x)$ in G (Claim 24). Wlog, take $y := y_0$, $y_i \equiv y^{p^i} \bmod p$, for all $i \in \{0, \ldots, b-1\}$ (Frobenius conjugates in $\mathbb{F}_p$). Note that $G \cong (\mathbb{Z}/\langle p^k\rangle)[y] =: G'$. We will prefer to use G' below.

The lemma below associates a root of f, in G or G', to a unique basic-irreducible factor of f in $(\mathbb{Z}/\langle p^k\rangle)[x]$.

▶ **Lemma 19** (Root to factor). *Let $r(y) \in G'$ be a root of $f(x)$. Then, $h(x) := \prod_{i=0}^{b-1}(x - r(y_i))$ is the unique basic-irreducible factor of f having root $r(y)$. We say: $h(x)$ is the basic-irreducible factor* associated to root $r(y)$.

Proof. The coefficients of h are symmetric polynomials in $r(y_i)$ (over $0 \le i < b$). Since the automorphism $\psi_1 : y \to y_1$ of G' (as defined in Claim 25) permutes $r(y_i)$'s ($\because$ it permutes y_i's), it fixes all the coefficients of h. From Claim 25, all these coefficients are then in $\mathbb{Z}/\langle p^k\rangle$. Hence, $h \in (\mathbb{Z}/\langle p^k\rangle)[x]$.

If $r(y)$ is a root of another polynomial h' in $(\mathbb{Z}/\langle p^k\rangle)[x]$, then $r(y_i)$'s are also roots of h' (applying automorphisms ψ_i of G'). Since these roots are coprime mod p, we actually get: $h|h'$. Thus, h is the unique monic irreducible factor of f containing $r(y)$.

Looking mod p, $r(y_i)$'s are a permutation of the roots of $\varphi(x)$, so $h(x) \equiv \varphi(x) \bmod p$. Hence, $h(x)$ is the unique monic basic-irreducible factor of f having root $r(y)$. ◀

Following is the reduction to counting all roots of f in G.

▶ **Theorem 20** (Factor to root). *Any degree-b basic-irreducible factor of $f \bmod p^k$ has exactly b roots in G. Conversely, if f has a root $r(y) \in G$, then it must be a root of a unique degree-b basic-irreducible factor of $f \bmod p^k$.*

So, the number of degree-b basic-irreducible factors of $f \bmod p^k$ is exactly the number of roots, of f in G, divided by the degree b.

Proof. By Lemma 19 (& uniqueness of Galois rings), for every root $r(y) \in G$ of f, we can associate a unique basic-irreducible factor of $f(x)$.

Conversely, let $h(x) =: \varphi(x) + pw(x)$ be a basic-irreducible factor of $f(x)$. It splits completely in G (as, $h(x) \equiv \varphi \bmod p$; first factor in $G/\langle p\rangle$ and then Hensel lift to G). So, h has exactly b roots in G, each of them is also a root of f in G.

Hence the theorem statement follows. ◀

Remark. This "irreducible factor vs root" correspondence, for $f \bmod p^k$, breaks down if G is *not* a Galois ring. E.g., what happens when the ring is $\mathbb{Z}[y]/\langle p^k, y^2 - p\rangle$?

4.2 Counting roots in $G(p^k, b)$– Wrapping up Thm. 2

In this section, we show how to count the roots of $f \equiv (\varphi_1\varphi_2 \ldots \varphi_t)^e + ph(x) \bmod p^k$ in $G(p^k, b)$. Since $G := G(p^k, b)$ is a Galois ring, so $G/\langle p\rangle = \mathbb{F}_{p^b} =: \mathbb{F}_q$. (Recall: $R = \mathbb{Z}/\langle p^k\rangle$.)

Split ideals and zerosets in the Galois ring. First, we will modify the definition of zerosets (Section 2) to include zeros of f in G. A *G-zeroset* of $f(x) \in R[x]$ will be defined as $\mathcal{Z}_G(f) := \{r \in G \mid f(r) \equiv 0 \bmod p^k\}$. Similarly, for an ideal $I \subseteq \mathbb{F}_p[\bar{x}_l]$, its $\mathbb{F}_q$-zeroset is defined as $\mathcal{Z}_{\mathbb{F}_q}(I) := \{\bar{a} = (a_0, \ldots, a_l) \in (\mathbb{F}_q)^{l+1} \mid g(\bar{a}) \equiv 0 \bmod p^k, \forall g \in I\}$.

The definition of triangular ideals, split ideals and maximal split ideals will remain exactly same (generators defined over $\mathbb{F}_p$, Section 2), except that in the third condition for split ideals, zeroset will be over $\mathbb{F}_q$ instead of $\mathbb{F}_p$. But, they can now be seen as storing potential roots of $f(x)$ in G (or, storing potential basic irreducible factors of $f \bmod p^k$). The reason is, a root

$r(y) \in G$ of $f \bmod p^k$ can be viewed as, $r(y) = r_0(y) + pr_1(y) + p^2r_2(y) + \ldots + p^{k-1}r_{k-1}(y)$, where each $r_i(y) \in G/\langle p\rangle = \mathbb{F}_q$. So, the decomposition of formal variable $x =: x_0 + px_1 + p^2x_2 + \ldots + p^{k-1}x_{k-1}$, now represents candidates for r_0, r_1, and so on, over $\mathbb{F}_q$.

A split ideal $I_l \subseteq \mathbb{F}_p[\bar{x}_l]$, defined as $I_l := \langle h_0(x_0), \ldots, h_l(\bar{x}_l)\rangle$, now implicitly stores the candidates for (r_0) in h_0, (r_0, r_1) in h_1, and so on. These, in turn, give candidates for basic-irreducible factors of $f \bmod p^{l'}$ (some $l' \leq k$).

In particular, when I_l is a maximal split ideal, an $\bar{r}_l$ implicitly denote a basic-irreducible factor of $f \bmod p^k$. The number of such factors is $\deg(I_l)\cdot q^{k-l-1}/b$ (Theorem 20 & Lemma 8).

Split ideals follow all the properties given in Section 2, just by replacing the fact that roots belong to $\mathbb{F}_q$ and not $\mathbb{F}_p$.

Description of the modified algorithm. Algorithm 1, to count roots in R, extends directly to count roots in G. The algorithm is exactly same except one change: to compute GCD (Steps 3 and 13), we now use the Frobenius polynomial $x^q - x$ instead of the prior $x^p - x$ (GCD computation implicitly stores the candidate roots, they are in $\mathbb{F}_q$ now).

So the algorithm works as follows:

1. It gets $f(x) \equiv (\varphi_1 \ldots \varphi_t)^e + pw(x) \bmod p^k$ as input, computes gcd $h_0(x) := \gcd(f(x), x^q - x)$ over $\mathbb{F}_p$. Since $x^q - x$, over $\mathbb{F}_p$, is the product of all irreducible factors of degree dividing b, we deduce: $h_0(x) = \varphi_1 \ldots \varphi_t \bmod p$; and define the first split ideal $I_0 := \langle h_0\rangle$. (Note– We do not have access to φ_i's themselves.)

   Remark. The length 1 split ideal stores all the roots of f in $G/\langle p\rangle$, or all the basic irreducible factors of $f \bmod p$; as $h_0(x) = \varphi_1 \ldots \varphi_t$. Also, its degree is tb, which when divided by b, gives the count of the basic-irreducible factors of $f \bmod p$.
2. The algorithm then successively looks for the next precision candidates. It computes h_l by taking gcd with $x^q - x$, and adds it to the previous ideal I_{l-1} like before.
3. All the supporting algebraic algorithms and lemmas (given in appendix) work the same as before; since they are being passed the same parameters – a split ideal, or a triangular ideal, or a polynomial over R.

Thus, a similar proof of correctness and time complexity can be given as before.

Proof of Theorem 2. Consider a univariate $f(x) \bmod p^k$. As discussed in the beginning of this section, $f \bmod p^k$ can be efficiently factorized as $f \equiv \prod_{i=1}^m f_i \bmod p^k$, where each $f_i(x)$ is a power of a product of degree-b_i irreducible polynomials mod p (i.e. of the form $\equiv (\varphi_1\varphi_2 \ldots \varphi_t)^e + ph(x)$, where φ_j is a degree-b_i irreducible mod p).

On each such $f_i \bmod p^k$, we use Algorithm 1 with the new Frobenius polynomial $(x^{q_i} - x)$ $(q_i = p^{b_i})$, in Steps 3 and 18, as discussed above. Let the final list output, for $f_i \bmod p^k$, be $\mathcal{L}_i =: \{I_1(l_1, D_1), \ldots, I_n(l_n, D_n)\}$. Thus, we get the count on the $G(p^k, b_i)$-roots of $f_i \bmod p^k$ as $\Sigma_{j=1}^n D_j q_i^{k-l_j}$ (Lemma 8). Using Theorem 20, the number of the degree-b_i basic-irreducible factors of $f \bmod p^k$ is $B_k(f_i) := (1/b_i) \times \Sigma_{j=1}^n D_j q_i^{k-l_j}$.

Using Lemma 22, we get the count on the basic-irreducible factors of $f \bmod p^k$ as, $B_k(f) = \Sigma_{i=1}^m B_k(f_i)$.

For the time complexity, only difference is the repeated-squaring to compute the reduced form of polynomial $x^{q_i} - x$ (Steps 3, 12), it will take $b_i \log p$ operations instead of $\log p$ operations. But $b_i \leq d$, so the algorithm runs in time poly$(d, k \log p)$ (& remains deterministic). ◀

5 Conclusion

There are well known efficient deterministic algorithms to count the number of roots/irreducible factors over prime characteristic. Surprisingly, not many results are known when the characteristic is a *prime-power*. The main difficulty is that the ring has *non*-unique factorization.

We give the first efficient deterministic algorithm to count the number of basic-irreducible factors modulo a prime-power. Restricting it to degree-one irreducibles, we get a deterministic polynomial-time algorithm to count the roots too. This is achieved by storing and improving roots (wrt precision) virtually using *split ideals* (we do not have access to roots directly). As a corollary: we can compute the Igusa zeta function deterministically, and we also get a deterministic algorithm to count roots in p-adic rings (resp. formal power-series ring).

Many interesting questions still remain to be tackled. For p-adic fields, there is only a randomized method to count the number of irreducible factors. Analogously, the question of counting irreducible factors modulo a prime-power also remains open; no efficient method is known even in the randomized setting. The *ramified* roots seem to elude practical methods. On the other hand, the problem of actually *finding* an irreducible factor (resp. a root) deterministically, seems much harder; it subsumes the analogous classic problem in prime characteristic.

References

1 Tom M Apostol. *Introduction to analytic number theory*. Springer Science & Business Media, 2013.

2 Jérémy Berthomieu, Grégoire Lecerf, and Guillaume Quintin. Polynomial root finding over local rings and application to error correcting codes. *Applicable Algebra in Engineering, Communication and Computing*, 24(6):413–443, 2013. https://link.springer.com/article/10.1007/s00200-013-0200-5.

3 Manjul Bhargava. P-orderings and polynomial functions on arbitrary subsets of Dedekind rings. *Journal fur die Reine und Angewandte Mathematik*, 490:101–128, 1997.

4 David G Cantor and Daniel M Gordon. Factoring Polynomials over p-Adic Fields. In *International Algorithmic Number Theory Symposium*, pages 185–208. Springer, 2000.

5 David G Cantor and Hans Zassenhaus. A new algorithm for factoring polynomials over finite fields. *Mathematics of Computation*, pages 587–592, 1981.

6 Qi Cheng, Shuhong Gao, J Maurice Rojas, and Daqing Wan. Counting Roots of Polynomials Over Prime Power Rings. In *Thirteenth Algorithmic Number Theory Symposium, ANTS-XIII*. Mathematical Sciences Publishers, 2018. `arXiv:1711.01355`.

7 AL Chistov. Efficient factorization of polynomials over local fields. *Dokl. Akad. Nauk SSSR*, 293(5):1073–1077, 1987.

8 AL Chistov. Algorithm of polynomial complexity for factoring polynomials over local fields. *Journal of mathematical sciences*, 70(4):1912–1933, 1994.

9 M Chojnacka-Pniewska. Sur les congruences aux racines données. In *Annales Polonici Mathematici*, volume 3, pages 9–12. Instytut Matematyczny Polskiej Akademii Nauk, 1956.

10 Bruce Dearden and Jerry Metzger. Roots of polynomials modulo prime powers. *European Journal of Combinatorics*, 18(6):601–606, 1997.

11 Jan Denef. Report on Igusa's local zeta function. *Astérisque*, 730-744(201-203):359–386, 1991.

12 Jan Denef and Kathleen Hoornaert. Newton polyhedra and Igusa's local zeta function. *Journal of number Theory*, 89(1):31–64, 2001.

13 Ashish Dwivedi, Rajat Mittal, and Nitin Saxena. Efficiently factoring polynomials modulo p^4. *The 44th International Symposium on Symbolic and Algebraic Computation (ISSAC)*, 2019. URL: `https://www.cse.iitk.ac.in/users/nitin/papers/factor-mod-p4.pdf`.

14 Andrzej Ehrenfeucht and Marek Karpinski. *The computational complexity of (xor, and)-counting problems*. International Computer Science Inst., 1990.

15 Kurt Hensel. Eine neue Theorie der algebraischen Zahlen. *Mathematische Zeitschrift*, 2(3):433–452, September 1918.

16 Jun-ichi Igusa. Complex powers and asymptotic expansions. I. Functions of certain types. *Journal für die reine und angewandte Mathematik*, 268:110–130, 1974.

17 Adam Klivans. Factoring polynomials modulo composites. Technical report, Carnegie-Mellon Univ, Pittsburgh PA, Dept of CS, 1997.

18 Neal Koblitz. P-adic numbers. In *p-adic Numbers, p-adic Analysis, and Zeta-Functions*, pages 1–20. Springer, 1977.

19 Leann Kopp, Natalie Randall, Joseph Rojas, and Yuyu Zhu. Randomized polynomial-time root counting in prime power rings. *Mathematics of Computation*, 2019. doi:10.1090/mcom/3431.

20 Alan GB Lauder. Counting solutions to equations in many variables over finite fields. *Foundations of Computational Mathematics*, 4(3):221–267, 2004.

21 Rudolf Lidl and Harald Niederreiter. *Introduction to finite fields and their applications*. Cambridge university press, 1994.

22 Davesh Maulik. Root sets of polynomials modulo prime powers. *Journal of Combinatorial Theory, Series A*, 93(1):125–140, 2001.

23 Bernard R McDonald. *Finite rings with identity*, volume 28. Marcel Dekker Incorporated, 1974.

24 Vincent Neiger, Johan Rosenkilde, and Éric Schost. Fast computation of the roots of polynomials over the ring of power series. In *Proceedings of the 2017 ACM on International Symposium on Symbolic and Algebraic Computation*, pages 349–356. ACM, 2017.

25 Ivan Niven, Herbert S Zuckerman, and Hugh L Montgomery. *An introduction to the theory of numbers*. John Wiley & Sons, 2013.

26 Ana Sălăgean. Factoring polynomials over $\mathbb{Z}_4$ and over certain Galois rings. *Finite fields and their applications*, 11(1):56–70, 2005.

27 Adi Shamir. On the generation of multivariate polynomials which are hard to factor. In *Proceedings of the twenty-fifth annual ACM symposium on Theory of computing*, pages 796–804. ACM, 1993.

28 Victor Shoup. *A computational introduction to number theory and algebra*. Cambridge university press, 2009.

29 Wacław Sierpiński. Remarques sur les racines d'une congruence. *Annales Polonici Mathematici*, 1(1):89–90, 1955.

30 Carlo Sircana. Factorization of Polynomials over $\mathbb{Z}/(p^n)$. In *Proceedings of the 2017 ACM on International Symposium on Symbolic and Algebraic Computation*, pages 405–412. ACM, 2017.

31 Joachim von zur Gathen and Silke Hartlieb. Factorization of polynomials modulo small prime powers. Technical report, Paderborn Univ, 1996.

32 Joachim von zur Gathen and Silke Hartlieb. Factoring modular polynomials. *Journal of Symbolic Computation*, 26(5):583–606, 1998. (Conference version in ISSAC'96).

33 Joachim von zur Gathen and Daniel Panario. Factoring polynomials over finite fields: A survey. *Journal of Symbolic Computation*, 31(1-2):3–17, 2001.

34 Hans Zassenhaus. On hensel factorization, I. *Journal of Number Theory*, 1(3):291–311, 1969.

35 WA Zuniga-Galindo. Computing Igusa's local zeta functions of univariate polynomials, and linear feedback shift registers. *Journal of Integer Sequences*, 6(2):3, 2003.

A Preliminaries

Lifting factorization. Below we state a lemma, originally due to Kurt Hensel [15], for $\mathcal{I}$-adic lifting of factorization of a given univariate polynomial. Over the years, Hensel's lemma has acquired many forms in different texts, version presented here is due to Zassenhaus [34].

▶ **Lemma 21** (Hensel's lemma [15]). *Let R be a commutative ring with unity, denote the polynomial ring over it by $R[x]$. Let $\mathcal{I} \subseteq R$ be an ideal of ring R. Given a polynomial $f(x) \in R[x]$, suppose f factorizes as*

$$f = gh \bmod \mathcal{I},$$

such that $gu + hv = 1 \bmod \mathcal{I}$ (for some $g, h, u, v \in R[x]$). Then, given any $l \in \mathbb{N}$, we can efficiently compute $g^*, h^*, u^*, v^* \in R[x]$, such that,

$$f = g^*h^* \bmod \mathcal{I}^l.$$

Here $g^* = g \bmod \mathcal{I}$, $h^* = h \bmod \mathcal{I}$ and $g^*u^* + h^*v^* = 1 \bmod \mathcal{I}^l$ (i.e. pseudo-*coprime lifts). Moreover g^* and h^* are unique up to multiplication by a unit.*

Using Hensel's lemma, for the purpose of counting roots (resp. basic-irreducible factors), a univariate polynomial $f(x) \in \mathbb{Z}[x]$ can be assumed to be a power of an irreducible modulo p.

▶ **Lemma 22.** *By the fundamental theorem of algebra, a univariate $f(x) \in \mathbb{Z}[x]$ factors uniquely, over $\mathbb{F}_p$, into coprime powers as, $f \equiv \prod_{i=1}^{m} {\varphi_i}^{e_i}$, where each $\varphi_i \in \mathbb{Z}[x]$ is irreducible mod p and $m, e_i \in \mathbb{N}$. Then, for all $k \in \mathbb{N}$,*

1. *f factorizes mod p^k as $f = g_1 g_2 \dots g_m$, where g_i's are mutually co-prime mod p^k and $g_i \equiv {\varphi_i}^{e_i} \bmod p$, for all $i \in [m]$.*
2. *any basic-irreducible factor of $f(x) \bmod p^k$ is a basic-irreducible factor of a unique $g_j \bmod p^k$, for some $j \in [m]$. Let $B_k(h)$ denote the number of (coprime) basic-irreducible factors of $h(x) \bmod p^k$. Then, $B_k(f) = \Sigma_{i=1}^{m} B_k(g_i)$.*
3. *any root of f mod p^k is a root of a unique g_i mod p^k. Let $N_k(h)$ denote the number of (distinct) roots of $h(x) \bmod p^k$. Then, $N_k(f) = \Sigma_{i=1}^{m} N_k(g_i)$.*

Proof. We can apply Hensel's lemma by taking ring $R := \mathbb{Z}$ and ideal $\mathcal{I} := \langle p \rangle$. The co-prime factorization of $f \bmod p$ lifts to a unique coprime factorization $f \equiv g_1 g_2 \dots g_m \bmod p^k$, for any $k \in \mathbb{N}$ and $g_i \equiv {\varphi_i}^{e_i} \bmod p$.

Any basic-irreducible factor $h(x)$ of $f(x) \bmod p^k$ has to be $h \equiv \varphi_i \bmod p$ for some $i \in [m]$; otherwise, h will become reducible mod p. Since g_i's are co-prime and $h | f \bmod p^k$, h must divide a unique g_i. So, any basic-irreducible factor h of $f(x) \bmod p^k$ is a basic-irreducible factor of a unique $g_j \bmod p^k$. Clearly, any basic-irreducible factor of a g_i is also a basic-irreducible factor of $f \bmod p^k$. This proves $B_k(f) = \Sigma_{i=1}^{m} B_k(g_i)$.

The third part follows from a similar reasoning as the second part. ◀

Root finding over a finite field: The following theorem, called CZ in this paper and given by Cantor-Zassenhaus [5], finds all roots of a given univariate polynomial over a finite field in randomized polynomial time. (Equivalently, it finds all irreducible factors as well.)

▶ **Theorem 23** (Cantor-Zassenhaus Algo (CZ)). *Given a univariate degree d polynomial $f(x)$ over a finite field $\mathbb{F}_q$, all roots of f in $\mathbb{F}_q$ can be found in randomized poly$(d, \log q)$ time.*

A.1 Properties of Galois rings– Analogues of finite fields

A *Galois ring*, of characteristic p^k and size p^{kb}, is denoted by $G(p^k, b)$ (where p is a prime, $k, b \in \mathbb{N}$). It is known that two Galois rings of same characteristic and size are isomorphic to each other. We will define Galois ring $G(p^k, b)$ as the ring $\mathbb{G} := \mathbb{Z}[y]/\langle p^k, \varphi(y) \rangle$, where $\varphi(y) \in \mathbb{Z}[y]$ is an irreducible mod p of degree b [23]. Let us prove some useful properties of $\mathbb{G}$ below.

▷ Claim 24 (Roots of φ). Let $\varphi'(x) \in \mathbb{Z}[x]$ be any irreducible mod p of degree b. There are b distinct roots of $\varphi'(x)$ in $\mathbb{G}$. Let r denote one of the roots, then all other roots, modulo p, are of the form r^{p^i} $(i \in \{0, \ldots, b-1\})$.

Proof. $\mathbb{G}/\langle p\rangle$ is isomorphic to the finite field of degree b over $\mathbb{F}_p$. So, irreducible $\varphi'(x) \in \mathbb{F}_p[x]$ has exactly b roots in $\mathbb{G}/\langle p\rangle$ [21, Ch.2]. By Hensel Lemma 21, roots in $\mathbb{G}/\langle p\rangle$ can be lifted to $\mathbb{G}$ uniquely. Hence, $\varphi'(x)$ has exactly b distinct roots in $\mathbb{G}$. Modulo p, they are of the form r^{p^i} $(i \in \{0, \ldots, b-1\})$ for a root r (lifted from roots in $\mathbb{G}/\langle p\rangle$). ◁

Using Claim 24, denote roots of $\varphi(x)$ as $y_0, \ldots, y_{b-1}$; here $y_i \equiv y_0^{p^i} \bmod p$ for all $i \in \{0, \ldots, b-1\}$. For all roots y_j, $\mathbb{G} \equiv R[y_j]$. In other words, y_j generate the extension $\mathbb{G}$ over R.

▷ Claim 25 (Symmetries of $\mathbb{G}$). There are exactly b automorphisms of $\mathbb{G}$ fixing $R = \mathbb{Z}/\langle p^k\rangle$, denoted by ψ_j $(j \in \{0, \ldots, b-1\})$. Each of these automorphisms can be described by a map taking y_0 to one of the roots of $\varphi(x)$ and fixing R. Wlog, assume ψ_j maps $y_0 \to y_j$.

Moreover, for all j coprime to b, ψ_j fixes R and nothing else.

Proof. Since coefficients of $\varphi(x)$ belong to R, an automorphism fixing R should map the root y_0 to another of its roots y_j. We only need to show that ψ_j is an automorphism (it is a valid map because $y_j \in \mathbb{G}$)

Writing elements of $\mathbb{G}$ in terms of y_0 (i.e. $\mathbb{G} \cong R[y_0]$), it can be verified that $\psi_j(ab) = \psi_j(a)\psi_j(b)$ and $\psi_j(a+b) = \psi_j(a) + \psi_j(b)$, so ψ_j is a homomorphism.

Similarly, if $\psi_j(g) = 0$, writing g in terms of y_0, we get that $g = 0$. So, kernel of ψ_j is the set $\{0\}$; thus, it is an isomorphism.

For the moreover part, let ψ_j be such that j is coprime to b. We will show a stronger statement by induction: for any $i \leq k-1$, if $a(y_0) = \psi_j(a(y_0))$ in $\mathbb{G}/\langle p^i\rangle$, then $a(y_0) \in \mathbb{Z}/\langle p^i\rangle$.

Base case: If $i = 1$ and $j = 1$, then $a(y_0) = \psi_1(a(y_0)) \bmod p \Rightarrow a(y_0) = a(y_0)^p \bmod p$. It means $a(y_0) \in \mathbb{Z}/\langle p\rangle$.

If j is coprime to b, then ψ_j generates ψ_1 modulo p. So, $a(y_0) = \psi_j(a(y_0)) \bmod p$ implies that, $a(y_0) \bmod p =: a_0 \in \mathbb{Z}/\langle p\rangle$.

This argument also proves: for any $i \leq k$, if $a(y_0) = a(y_j)$ in $\mathbb{G}/\langle p^i\rangle$, then $a(y_0) \in \mathbb{F}_p$ (in other words, $a(y_0)$ is y_0 free).

Induction step: Let us assume that $a(y_0) = \psi_j(a(y_0))$ in $\mathbb{G}/\langle p^i\rangle$. By the previous argument, $a(y_0) = a_0 + pa'(y_0)$, where $a_0 \in \mathbb{Z}/\langle p\rangle$ and $a'(y_0) \in \mathbb{G}/\langle p^{i-1}\rangle$.

From the definition, $a(y_0) = \psi_j(a(y_0))$ iff $a'(y_0) = \psi_j(a'(y_0))$ in $\mathbb{G}/\langle p^{i-1}\rangle$. By induction hypothesis, the latter is equivalent to $a'(y_0) \in \mathbb{Z}/\langle p^{i-1}\rangle$. So, $a(y_0) \in \mathbb{Z}/\langle p^i\rangle$.

Hence, the only fixed elements under the map ψ_j (j coprime to b) are integers; in $\mathbb{Z}/\langle p^k\rangle$. ◁

B Proofs of Section 2

Proof of Lemma 5. It is enough to show the lemma for $j = l-1$. It is easy to observe that I_{l-1} is triangular.

Looking at the second condition for being a split ideal, $|\mathcal{Z}_{\mathbb{F}_p}(I_{l-1})| \leq \prod_{i=0}^{l-1} \deg_{x_i}(h_i)$ follows because a degree $d \geq 1$ polynomial can have at most d roots in $\mathbb{F}_p$.

To show equality, notice that for any $\bar{a} = (a_0, \ldots, a_{l-1}) \in \mathcal{Z}_{\mathbb{F}_p}(I_{l-1})$, $\deg_{x_l}(h_l(\bar{a}, x_l))$ is bounded by $\deg_{x_l}(h_l)$. This implies $h_l(\bar{a}, x_l)$ can have at most $\deg_{x_l}(h_l)$ roots in $\mathbb{F}_p$. If

$|\mathcal{Z}_{\mathbb{F}_p}(I_{l-1})| < \prod_{i=0}^{l-1} \deg_{x_i}(h_i)$ then $|\mathcal{Z}_{\mathbb{F}_p}(I_l)| < \deg_{x_l}(h_l) \cdot \prod_{i=0}^{l-1} \deg_{x_i}(h_i)$, contradicting that I_l is a split ideal. [3]

For the third condition, since I_l is a split ideal, for any $(a_0, \ldots, a_{l-1}) \in \mathcal{Z}_{\mathbb{F}_p}(I_{l-1})$, $f(a_0 + pa_1 + \ldots + p^l a_l) \equiv 0 \bmod p^{l+1} \Rightarrow f(a_0 + pa_1 + \ldots + p^{l-1}a_{l-1}) \equiv 0 \bmod p^l$. ◀

Lemma 6 shows that a split ideal I can be decomposed in terms of ideals $I_{\bar{a}} := \langle x_0 - a_0, \ldots, x_l - a_l \rangle$, where $\bar{a} =: (a_0, \ldots, a_l)$ is a root of I. Before we prove this structural lemma, let us see some properties of these ideals $I_{\bar{a}}$'s.

▷ Claim 26. Let I be a split ideal.

1. For any ideal $I_{\bar{a}}$, quotient $\mathbb{F}_p[x_0, \ldots, x_l]/I_{\bar{a}} \cong \mathbb{F}_p$ is a *field*.
2. $I_{\bar{a}}$ and $I_{\bar{b}}$ are *coprime* for any two distinct roots $\bar{a}, \bar{b} \in \mathcal{Z}_{\mathbb{F}_p}(I)$. This is because there exists i, for which $a_i \neq b_i$; yielding $(a_i - b_i)^{-1}((x_i - b_i) - (x_i - a_i)) = 1$ in the sum-ideal $I_{\bar{a}} + I_{\bar{b}}$.
3. $I_{\bar{a}} \cap I_{\bar{b}} = I_{\bar{a}} I_{\bar{b}}$ for any two distinct roots $\bar{a}, \bar{b} \in \mathcal{Z}_{\mathbb{F}_p}(I)$. It follows because there exist $r_{\bar{a}} \in I_{\bar{a}}$ and $r_{\bar{b}} \in I_{\bar{b}}$, s.t., $r_{\bar{a}} + r_{\bar{b}} = 1$. So, $r \in I_{\bar{a}} \cap I_{\bar{b}} \Rightarrow r = r(r_{\bar{a}} + r_{\bar{b}}) \in I_{\bar{a}} I_{\bar{b}}$. On the other hand, $I_{\bar{a}} I_{\bar{b}} \subseteq I_{\bar{a}} \cap I_{\bar{b}}$ follows from the definition of the product-ideal.
4. Generalizing the previous point – for a set A of distinct roots $\bar{a}$'s, $\bigcap_{\bar{a} \in A} I_{\bar{a}} = \prod_{\bar{a} \in A} I_{\bar{a}}$.

Proof of Lemma 6. We will prove this decomposition by applying induction on the length of the split ideal. For the base case, length of I is 1 and $I = \langle h_0(\bar{x}_0) \rangle \subseteq \mathbb{F}_p[x_0]$. Since I is a split ideal, $h_0(x_0) = \prod_{i=1}^{\deg(h_0)} (x_0 - a_i)$ for *distinct* $a_i \in \mathbb{F}_p$. So, $I = \prod_{i=1}^{\deg(h_0)} I_{a_i} = \bigcap_{i=1}^{\deg(h_0)} I_{a_i}$ by Claim 26.

Let I be a split ideal of length $l+1$, $I =: \langle h_0(\bar{x}_0), \ldots, h_l(\bar{x}_l) \rangle \subseteq \mathbb{F}_p[x_0, \ldots, x_l]$. Define ideal $I' := \langle h_0(\bar{x}_0), \ldots, h_{l-1}(\bar{x}_{l-1}) \rangle$. By Lemma 5, I' is a split ideal. From the induction hypothesis (& Claim 26), we have $I' = \bigcap_{\bar{a} \in \mathcal{Z}_{\mathbb{F}_p}(I)} I'_{\bar{a}} = \prod_{\bar{a}} I'_{\bar{a}}$, where $I'_{\bar{a}} := \langle x_0 - a_0, \ldots, x_{l-1} - a_{l-1} \rangle$ for a zero $\bar{a} =: (a_0, \ldots, a_{l-1})$ of I'. We know that,

$$I \;=\; I' + \langle h_l(\bar{x}_l) \rangle \;=\; \prod_{\bar{a} \in \mathcal{Z}_{\mathbb{F}_p}(I')} \left(I'_{\bar{a}} + \langle h_l(\bar{x}_l) \rangle \right) . \tag{4}$$

Claim 10 shows $\deg(h_l(\bar{a}, x_l)) = \deg_{x_l}(h_l)$ for all $\bar{a} \in \mathcal{Z}_{\mathbb{F}_p}(I')$, and $h_l(\bar{a}, x_l)$ splits completely over $\mathbb{F}_p$. So, for any $\bar{a} \in \mathcal{Z}_{\mathbb{F}_p}(I')$, $I'_{\bar{a}} + \langle h_l(\bar{x}_l) \rangle = \prod_{i=1}^{\deg_{x_l}(h_l)} I_{\bar{a}, b_i}$, where $(\bar{a}, b_i)$ are roots of I extended from $\bar{a}$. From Eqn. 4 (& Claim 26), $I = \prod_{\bar{b} \in \mathcal{Z}_{\mathbb{F}_p}(I)} I_{\bar{b}} = \bigcap_{\bar{b} \in \mathcal{Z}_{\mathbb{F}_p}(I)} I_{\bar{b}}$.

This finishes the inductive proof, completely factoring I. ◀

Lemma 8 shows that a root of a maximal split ideal represents a set of roots of $f \bmod p^k$ and provides the size of that set.

Proof of Lemma 8. By definition of a maximal split ideal, for any $\bar{a} = (a_0, \ldots, a_l) \in \mathcal{Z}_{\mathbb{F}_p}(I)$, $p^k | g(x)$ where $g(x) = f(a_0 + pa_1 + p^2 a_2 + \ldots + p^l a_l + p^{l+1} x)$. So, $g(x) = 0 \bmod p^k$ for any p^{k-l-1} choices of x. For each such fixing of x, $a_0 + pa_1 + p^2 a_2 + \ldots + p^l a_l + p^{l+1} x$ is a *distinct* root of $f(x) \bmod p^k$. Hence proved. ◀

[3] This argument also shows that every $\mathbb{F}_p$-zero of I_{l-1} "extends" to exactly $\deg_{x_l}(h_l)$ many $\mathbb{F}_p$-zeros of I_l.

C Computation modulo a triangular ideal– Reduce & Divide

For completeness, we show that it is efficient to reduce a polynomial $a(\bar{x}_l) \in \mathbb{G}[\bar{x}_l]$ modulo a triangular ideal $J_l = \langle b_0(\bar{x}_0), b_1(\bar{x}_1), \ldots, b_l(\bar{x}_l)\rangle \subseteq \mathbb{G}[\bar{x}_l]$, where $\mathbb{G}$ is any Galois ring (in particular, $R = \mathbb{Z}/p^k$, or $\mathbb{F}_p$).

Note: J_l need not be a split ideal for $f \bmod p^k$, though the algorithms of this section work for split ideals ($\because$ they are triangular by definition).

Assumptions: In the generators of the triangular ideal we assume $\deg_{x_i} b_i(\bar{x}_i) \geq 2$ (for $0 \leq i \leq l$). Otherwise, we could eliminate variable x_i and work with fewer variables (& smaller length triangular ideal). Additionally, each $b_i(\bar{x}_i)$ (for $0 \leq i \leq l$) is monic (leading coefficient is 1 wrt x_i), and presented in a *reduced* form modulo the prior triangular ideal $J_{i-1} := \langle b_0(\bar{x}_0), \ldots, b_{i-1}(\bar{x}_{i-1})\rangle \subseteq \mathbb{G}[\bar{x}_{i-1}]$.

Let us first define reduction mod an ideal (assume $\mathbb{G}$ to be the Galois ring $G(p^k, b)$).

▶ **Definition 27** (Reduction by a triangular ideal)**.** *The* reduction *of a multivariate polynomial $a(\bar{x}_l) \in \mathbb{G}[\bar{x}_l]$ by a triangular ideal $J_l = \langle b_0(\bar{x}_0), \ldots, b_l(\bar{x}_l)\rangle \subseteq \mathbb{G}[\bar{x}_l]$ is the unique polynomial $\tilde{a}(\bar{x}_l) \equiv a(\bar{x}_l) \bmod J_l$, where $\deg_{x_i}(\tilde{a}) < \deg_{x_i}(b_i)$, for all $i \in \{0, \ldots, l\}$.*

Idea of reduction. The idea behind the algorithm is inspired from the univariate reduction. If $l = 0$, then reduction of $a(x_0)$ modulo $b_0(x_0)$ is simply the remainder of the division of a by b_0 in the underlying polynomial ring $\mathbb{G}[x_0]$. For a larger l, the reduction of $a(\bar{x}_l)$ modulo the triangular ideal $J_l = \langle b_0(x_0), \ldots, b_l(\bar{x}_l)\rangle$ is the remainder of the division of $a(\bar{x}_l)$ by $b_l(\bar{x}_l)$ in the polynomial ring $(\mathbb{G}[x_0, \ldots, x_{l-1}]/J_{l-1})[x_l]$. The fact that b_l is monic, helps in generalizing "long division".

Input: An $a(\bar{x}_l) \in \mathbb{G}[\bar{x}_l]$ and a triangular ideal $J_l = \langle b_0(\bar{x}_0), \ldots, b_l(\bar{x}_l)\rangle \subseteq \mathbb{G}[\bar{x}_l]$.

Output: Reduction $\tilde{a}$ of a mod J_l as defined above.

Algorithm 2 Reduce $a(\bar{x}_l)$ modulo J_l.

1: **procedure** REDUCE($a(\bar{x}_l)$, J_l)
2: **if** $l = 0$ **then**
3: [**Reduce** $a(x_0)$ **by** $b_0(x_0)$] return remainder of univariate division of a by b_0 in $R[x_0]$.
4: **end if**
5: $d_a \leftarrow \deg_{x_l}(a)$ and $d_b \leftarrow \deg_{x_l}(b_l)$.
6: Let $a(\bar{x}_l) =: \Sigma_{i=0}^{d_a} a_i(\bar{x}_{l-1}) x_l^i$ be the polynomial representation of $a(\bar{x}_l)$ with respect to x_l.
7: Recursively reduce each coefficient $a_i(\bar{x}_{l-1})$ of $a \bmod J_{l-1}$:
$\tilde{a}_i(\bar{x}_{l-1}) \leftarrow$ REDUCE($a_i(\bar{x}_{l-1})$, J_{l-1}), for all $i \in \{0, \ldots, d_a\}$.
8: **while** $d_a \geq d_b$ **do**
9: $a(\bar{x}_l) \leftarrow a - \left(a_{d_a} \cdot x_l^{d_a - d_b} \cdot b_l\right)$
10: Update $d_a \leftarrow \deg_{x_l}(a)$. Update a_i's such that $a(\bar{x}_l) =: \Sigma_{i=0}^{d_a} a_i(\bar{x}_{l-1}) \cdot x_l^i$.
11: Call REDUCE($a_i(\bar{x}_{l-1})$, J_{l-1}) for all $i \in \{0, \ldots, d_a\}$: recursively reduce each coefficient $a_i(\bar{x}_{l-1}) \bmod J_{l-1}$ (like Step 7).
12: **end while**
13: **return** $a(\bar{x}_l)$.
14: **end procedure**

Following lemma shows that reduction modulo a triangular ideal (Algorithm 2) is efficient.

▶ **Lemma 28** (Reduction). *Given $a(\bar{x}_l) \in \mathbb{G}[\bar{x}_l]$ and $J_l \subseteq \mathbb{G}[\bar{x}_l]$, to reduce $a(\bar{x}_l)$ mod J_l, Algorithm 2 takes time poly $\left(\prod_{i=0}^{l} \deg_{x_i}(a), \log|\mathbb{G}|, \deg(J_l)\right)$.*

In particular, if each coefficient $a_i(\bar{x}_{l-1})$ of $a(\bar{x}_l)$ (viewed as a polynomial in x_l) is in reduced form mod J_{l-1}, then reduction takes time poly$(d_a, \log|\mathbb{G}|, \deg(J_l))$, where $d_a = \deg_{x_l}(a)$.

Proof. We prove the lemma by induction on the length $l+1$ of the ideal J_l.

For $l = 0$, we have a standard univariate reduction which takes at most $O(\deg(a)\deg(b))$ ring operations in $\mathbb{G}$. Since addition/multiplication/division in $\mathbb{G}$ take time at most $\tilde{O}(\log|\mathbb{G}|)$ [28], we get the lemma.

Assume that the lemma is true for any ideal of length less than l.

Coefficients $a_i(\bar{x}_{l-1})$ can be reduced, in time poly $\left(\prod_{i=0}^{l-1} \deg_{x_i}(a), \log|\mathbb{G}|, \deg(J_{l-1})\right)$, mod J_{l-1} using induction hypothesis. We need to make $d_a + 1$ such calls; total time is bounded by poly $\left(\prod_{i=0}^{l} \deg_{x_i}(a), \log|\mathbb{G}|, \deg(J_{l-1})\right)$. In the same time we can compute Step 9.

After the update at Step 9, individual-degrees $\deg_{x_i}(a)$ (for $0 \le i < l$) can become at most double the previous degree (safely assuming $2 \le \deg_{x_i}(b_i) \le \deg_{x_i}(a)$). By induction hypothesis, each call to reduce $a_i(\bar{x}_{l-1})$ mod J_{l-1} takes time poly $\left(\prod_{i=0}^{l-1} \deg_{x_i}(a), \log|\mathbb{G}|, \deg(J_{l-1})\right)$. Algorithm makes at most d_a such calls and the while-loop runs at most d_a times. Hence, the algorithm takes time poly $\left(\prod_{i=0}^{l} \deg_{x_i}(a), \log|\mathbb{G}|, \deg(J_l)\right)$; and we are done.

If coefficients of a are already reduced modulo J_{l-1}, then $\deg_{x_i}(a) < \deg_{x_i}(b_i)$ for all $0 \le i < l$. Hence, Algorithm 2 takes time $d_a^2 \cdot$ poly $(\log|\mathbb{G}|, \deg(J_{l-1}))$. ◀

▶ **Lemma 29** (Division mod triangular ideal). *Given a triangular ideal $J_l \subseteq \mathbb{G}[\bar{x}_l]$ and a unit $a(\bar{x}_l) \in \mathbb{G}[\bar{x}_l]/J_l$. We can compute $a^{-1} \bmod J_l$, in reduced form, in time poly $\left(\prod_{i=0}^{l} \deg_{x_i}(a), \log|\mathbb{G}|, \deg(J_l)\right)$.*

Proof. Let $u(\bar{x}_l) \in \mathbb{G}[\bar{x}_l]/J_l$ be such that $u \cdot a \equiv 1 \bmod J_l$. We can write u as

$$\sum_{\substack{\bar{e} \ge \bar{0} \\ \forall\ 0 \le i \le l,\ e_i < \deg_{x_i}(b_i)}} u_{\bar{e}} \cdot \bar{x}_l^{\bar{e}} \ .$$

We want to find the unknowns $u_{\bar{e}}$ in $\mathbb{G}$, satisfying $u \cdot a \equiv 1 \bmod J_l$. This gives us a linear system in the unknowns; it has size $\deg(J_l)$. The linear system can be written down, using Algorithm 2, by reducing the monomial products $\bar{x}_l^{\bar{e}} \cdot \bar{x}_l^{\bar{e}'}$ that appear in the product $u \cdot a$. This takes time poly $\left(\prod_{i=0}^{l} \deg_{x_i}(a), \log|\mathbb{G}|, \deg(J_l)\right)$.

Since there exists a unique u, our linear system is efficiently solvable, by standard linear algebra, in the required time. ◀

Let us see two direct applications of the reduction Algorithm 2 to compute valuation and to compute reduced form of split ideals.

First, we explain how Algorithm 1 (Steps 3, 18) computes reduced f_J modulo the lift $\hat{J}$ of the newly computed split ideal J, when x is replaced by $x_{l+1} + px$ in the intermediate polynomial $f_I(\bar{x}_l, x)$.

▶ **Lemma 30** (Updating stack with reduced polynomial). *Let $I \subseteq \mathbb{F}_p[\bar{x}_l]$ be a split ideal and $f_I(\bar{x}_l, x) \in R[\bar{x}_l, x]$ be reduced modulo $\hat{I}$ (the lift of I over R). Define split ideal $J \subseteq \mathbb{F}_p[\bar{x}_{l+1}]$ as $J := I + \langle h_{l+1}(\bar{x}_{l+1})\rangle$, and $\hat{J}$ be the lift of J over R.*

Then, in time poly$(\log|R|, \deg_x(f_I), \deg(J))$, we can compute a reduced polynomial f_J modulo $\hat{J}$ defined by, $f_J(\bar{x}_{l+1}, x) := f_I(\bar{x}_l, x_{l+1} + px) \bmod \hat{J}$.

Proof. Since $f_I(\bar{x}_l, x)$ is already reduced modulo $\hat{I}$, $\deg_{x_i}(f_I) < \deg_{x_i}(h_i)$. Define $D := \deg_x(f_I)$, perform the shift $x \to x_{l+1} + px$ in f_I, and expand f_I using Taylor series,

$$f_J(\bar{x}_l, x) \;=\; f_I(\bar{x}_l, x_{l+1} + px) \;=:\; g_0(\bar{x}_{l+1}) + g_1(\bar{x}_{l+1})(px) + \ldots + g_D(\bar{x}_{l+1})(px)^D \;,$$

where g_i could also be seen as the i-th derivative of $f_I(\bar{x}_l, x_{l+1})$ (wrt x_{l+1}) divided by $i!$. To compute f_J mod $\hat{J}$, we call $\textsc{Reduce}(g_i, \hat{J})$ (for all i) to get the reduction of each term mod $\hat{J}$.

To calculate the time complexity of $\textsc{Reduce}(g_i, \hat{J})$, note that coefficients of each g_i, wrt x_{l+1}, is already reduced mod $\hat{I}$. Since $J = I + \langle h_{l+1} \rangle$, using Lemma 28, time complexity of reducing each g_i by $\hat{J}$ is at most $\text{poly}(\deg_{x_{l+1}}(g_i), \log|R|, \deg(J))$ ($\deg(J) = \deg(\hat{J})$).

Since $\deg_{x_{l+1}}(g_i) \leq \deg_x(f_I)$ (for $i \leq D$), total time complexity is $\text{poly}(\log|R|, \deg_x(f_I), \deg(J))$. ◀

Next, we explain Step 20 in Algorithm 1 a bit more.

▶ **Lemma 31** (Ideal factors in reduced form). *Consider the tuple* $(U := \{h_0(\bar{x}_0), \ldots, h_l(\bar{x}_l)\}, f_{\langle U \rangle}) \in S$ *and consider a non-trivial factorization* $h_i =: h_{i,1} \ldots h_{i,m}$ *for some* $h_i \in U$. *Wlog each factor* $h_{i,j}$ *is monic wrt* x_i.

Then, we can compute the factor-related tuples $(U_j, f_{\langle U_j \rangle})$, *for all* $j \in [m]$, *in time* $\text{poly}(\deg(\langle U \rangle), \log|R|, \deg_x(f_{\langle U \rangle}))$ ($f_{\langle U_j \rangle}$ *will be in reduced form mod* $\langle U_j \rangle$).

Proof. First, we successively reduce h_{i+t} ($1 \leq t \leq l - i$) modulo triangular ideal $I_{i+t,j} := \langle h_0, \ldots, h_{i-1}, h_{i,j}, h_{i+1}, \ldots, h_{i+t} \rangle$. Time complexity of each of these steps is bounded by $\text{poly}(\deg(\langle U \rangle), \log|R|)$ (Lemma 28). This ensures that the degree of h_{i+t} in a variable x_s ($s < i + t$) is less than the individual-degree of the s-th generator of ideal $\langle U_j \rangle$.

Then, $f_{\langle U_j \rangle}$ can be calculated by reducing each $\deg_x(f_{\langle U \rangle})+1$ coefficients of $f_{\langle U_j \rangle}$ (wrt x) by the lifted triangular ideal $\hat{I}_{l,j} = \hat{U}_j$. By Lemma 28, this takes time $\text{poly}(\prod_{i=0}^{l} \deg_{x_i}(f_{\langle U \rangle}), \deg_x(f_{\langle U \rangle}), \log|R|, \deg(\langle U \rangle))$. Since coefficients (wrt x) of $f_{\langle U \rangle}$ were already reduced modulo $\langle U \rangle$, $\prod_{i=0}^{l} \deg_{x_i}(f_{\langle U \rangle}) \leq \deg(\langle U \rangle)$.

So, the computation time is bounded by $\text{poly}\left(\deg(\langle U \rangle), \log|R|, \deg_x(f_{\langle U \rangle})\right)$. ◀

D Computation modulo a triangular ideal – Zerodivisor test & GCD

$\textsc{Test-Zero-Div}(a(\bar{x}_l), I_l)$, for a triangular ideal $I_l =: \langle h_0, \ldots, h_l \rangle$, either reports that $a(\bar{x}_l)$ is not a zerodivisor modulo I_l, or *returns a non-trivial factorization* of a generator $h_i =: h_{i,1} \cdots h_{i,m}$ (into monic, wrt x_i, factors mod prior ideal). In this section we assume $\mathbb{F}$ to be a finite field.

Idea. In the quotient ring $\mathbb{F}[\bar{x}_l]/\langle I_l \rangle$, a monic (wrt x_i) polynomial $a(\bar{x}_i)$ is a zerodivisor iff it contains a factor of $h_i(\bar{x}_i)$ – generator of triangular ideal I_l with variables $\{x_0, \ldots, x_i\}$. So, firstly the algorithm checks if the given polynomial $a(\bar{x}_l)$ is monic (recursively, from variables x_{l-1} to x_0). If it fails, it factors some generator h_i for $i < l$. After making $a(\bar{x}_l)$ monic, we take gcd of a with h_l – if it finds non-trivial gcd it factors h_l, else $a(\bar{x}_l)$ is not a zerodivisor.

▶ **Lemma 32** (Efficiency of testing zerodivisors). *Assuming, coefficients of* $a(\bar{x}_l)$ *wrt* x_l *are in reduced form modulo* I_{l-1}, *Algorithm 3 takes time* $\text{poly}(\deg_{x_l}(a), \log|\mathbb{F}|, \deg(I_l))$.

Proof. We apply induction on the length $l + 1$ of ideal I_l.

For $l = 0$, it runs univariate gcd and takes time $\text{poly}(\deg(a), \deg(h_0), \log|\mathbb{F}|)$ [28].

Assume lemma statement holds true for ideals of length l.

Algorithm 3 Zerodivisor test of $a(\bar{x}_l)$ modulo I_l.

procedure TEST-ZERO-DIV($a(\bar{x}_l)$,I_l)
  if $l = 0$ **then**
    [**Take univariate GCD**] $gcd \leftarrow \gcd(a(x_0), h_0(x_0))$.
    if gcd is non-trivial **then**
      Factorize $h_0(x_0) =: gcd \cdot \frac{h_0}{gcd}$; **return** $(True, gcd \cdot \frac{h_0}{gcd})$.
    else
      return $(False)$.
    end if
  end if
  Let the leading coefficient of $a(\bar{x}_l)$ wrt x_l be $\tilde{a}(\bar{x}_{l-1})$.
  Call TEST-ZERO-DIV($\tilde{a}(\bar{x}_{l-1})$, I_{l-1}).
  if The test returned $True$ **then**
    return the result of the test including the factorization of a generator $h_i(\bar{x}_i)$.
  end if
  [Now, we will take gcd of a and h_l using iterated division method (Euclid's method).]
  Define $b(\bar{x}_l) \leftarrow h_l(\bar{x}_l)$.
  while $b(\bar{x}_l) \neq 0$ **do**
    Let $\tilde{b}(\bar{x}_{l-1})$ be the leading coefficient of $b(\bar{x}_l)$ wrt x_l.
    if TEST-ZERO-DIV($\tilde{b}(\bar{x}_{l-1})$, I_{l-1}) $= True$ **then**
      return result of TEST-ZERO-DIV($\tilde{b}(\bar{x}_{l-1})$,I_{l-1}), factorization of a generator $h_i(\bar{x}_i)$.
    end if
    Let $c(\bar{x}_l) \leftarrow$ REDUCE($a(\bar{x}_l)$, $I_{l-1} + \langle b(\bar{x}_l)/\tilde{b}\rangle$) (same as taking remainder of $a(\bar{x}_l)$ when divided by the monic polynomial $b(\bar{x}_l)/\tilde{b}$ modulo I_{l-1}).
    $a(\bar{x}_l) \leftarrow b(\bar{x}_l)/\tilde{b}$, $b(\bar{x}_l) \leftarrow c(\bar{x}_l)$. [Invariant: $\deg_{x_l}(b)$ has fallen.]
  end while
  [Gcd of original $a(\bar{x}_l)$ and $h_l(\bar{x}_l)$ mod I_l is stored in $a(\bar{x}_l)$.]
  if gcd $a(\bar{x}_l)$ is non-trivial **then**
    return ($True$, a non-trivial factorization of $h_l(\bar{x}_l)$).
  else
    return $(False)$. [$a(\bar{x}_l)$ is not a zerodivisor.]
  end if
end procedure

By induction, checking $\tilde{a}(\bar{x}_{l-1})$ is a zerodivisor mod I_{l-1}, takes $\text{poly}(\deg_{x_{l-1}}(\tilde{a}), \log|\mathbb{F}|, \deg(I_{l-1}))$ time.

To compute gcd of a and h_l, Euclidean gcd algorithm will run at most $\deg_{x_l}(a) + \deg_{x_l}(h_l)$ while-loops. From induction hypothesis, and Lemmas 28-29, each loop takes at most $\text{poly}(\deg_{x_l}(a), \log|\mathbb{F}|, \deg(I_l))$ time. So, we are done. ◀

GCD($a(\bar{x}_l, x)$, $b(\bar{x}_l, x)$, I_l) computes gcd of two polynomials $a(\bar{x}_l, x)$ and $b(\bar{x}_l, x)$ modulo a triangular ideal $I_l = \langle h_0(x_0), \ldots, h_l(\bar{x}_l)\rangle$ resp. $False$. It computes the *monic gcd* resp. returns a *non-trivial factorization* of some h_i.

Algorithm 4 GCD computation modulo I_l.

procedure GCD($a(\bar{x}_l, x)$, $b(\bar{x}_l, x)$, I_l)
  Let $\tilde{b}(\bar{x}_l)$ be the leading coefficient of b with respect to x.
  if Test-Zero-Div($\tilde{b}(\bar{x}_l)$, I_l) $= True$ **then**
    return $False$, Test-Zero-Div($\tilde{b}(\bar{x}_l)$, I_l) factors some generator $h_i(\bar{x}_i)$.
  end if
  Let $c(\bar{x}_l, x) \leftarrow$Reduce($a$, $I_l + \langle b/\tilde{b} \rangle$).
  if $c = 0$ **then**
    return $b/\tilde{b}$.
  else
    return GCD($b(\bar{x}_l, x)$, $c(\bar{x}_l, x)$, I_l).
  end if
end procedure

▶ **Lemma 33** (Multivariate GCD). *Algorithm 4 either factors a generator h_i (& outputs $False$), or computes a monic polynomial $g(\bar{x}_l, x) \in \mathbb{F}[\bar{x}_l, x]$, such that, g divides a, b modulo I_l. Moreover, $g = ua + vb \bmod I_l$, for some $u(\bar{x}_l, x), v(\bar{x}_l, x) \in \mathbb{F}[\bar{x}_l, x]$.*

If a and b are in reduced form mod I_l, then it takes time poly$(\deg_x(a), \deg_x(b), \log|\mathbb{F}|, \deg(I_l))$.

Proof. Algorithm 4 is just an implementation of multivariate Euclidean gcd algorithm over the coefficient ring $\mathbb{F}_p[\bar{x}_l]/I_l =: R'$. If the algorithm outputs $g(\bar{x}_l, x) \in R'[x_l]$ then, by standard Euclidean gcd arguments (using recursion), there exists $u(\bar{x}_l, x), v(\bar{x}_l, x) \in R'[x]$, such that, $ua + vb = g$, and g divides both a and b modulo I_l.

The algorithm works fine if in each step it was able to work with a monic divisor. Otherwise, it gets stuck at a "division" step, implying that the divisor's leading-coefficient is a zerodivisor, factoring some generator of I_l.

For time complexity, each recursive step makes one call each to Test-Zero-Div, Reduce, and division procedures. They take time poly$(\deg_x(a), \deg_x(b), \log|\mathbb{F}|, \deg(I_l))$ ($\because$ coefficients of a and b are in reduced form mod I_l, and use Lemmas 28, 29 & 32). Since number of recursive steps are bounded by $\deg_x(a) + \deg_x(b)$, total time is bounded by poly$(\deg_x(a), \deg_x(b), \log|\mathbb{F}|, \deg(I_l))$. ◀

Near-Optimal Pseudorandom Generators for Constant-Depth Read-Once Formulas

Dean Doron
Department of Computer Science, University of Texas at Austin, USA
deandoron@utexas.edu

Pooya Hatami
Department of Computer Science, University of Texas at Austin, USA
https://pooyahatami.org
pooyahat@gmail.com

William M. Hoza
Department of Computer Science, University of Texas at Austin, USA
https://williamhoza.com
whoza@utexas.edu

Abstract

We give an explicit pseudorandom generator (PRG) for read-once $\mathbf{AC}^0$, i.e., constant-depth read-once formulas over the basis $\{\wedge, \vee, \neg\}$ with unbounded fan-in. The seed length of our PRG is $\widetilde{O}(\log(n/\varepsilon))$. Previously, PRGs with near-optimal seed length were known only for the depth-2 case [22]. For a constant depth $d > 2$, the best prior PRG is a recent construction by Forbes and Kelley with seed length $\widetilde{O}(\log^2 n + \log n \log(1/\varepsilon))$ for the more general model of constant-width read-once branching programs with arbitrary variable order [17]. Looking beyond read-once $\mathbf{AC}^0$, we also show that our PRG fools read-once $\mathbf{AC}^0[\oplus]$ with seed length $\widetilde{O}(t + \log(n/\varepsilon))$, where t is the number of parity gates in the formula.

Our construction follows Ajtai and Wigderson's approach of iterated pseudorandom restrictions [1]. We assume by recursion that we already have a PRG for depth-d $\mathbf{AC}^0$ formulas. To fool depth-$(d+1)$ $\mathbf{AC}^0$ formulas, we use the given PRG, combined with a small-bias distribution and almost k-wise independence, to sample a pseudorandom restriction. The analysis of Forbes and Kelley [17] shows that our restriction approximately preserves the expectation of the formula. The crux of our work is showing that after $\mathrm{poly}(\log \log n)$ independent applications of our pseudorandom restriction, the formula simplifies in the sense that every gate other than the output has only $\mathrm{polylog}\, n$ remaining children. Finally, as the last step, we use a recent PRG by Meka, Reingold, and Tal [32] to fool this simpler formula.

2012 ACM Subject Classification Theory of computation → Pseudorandomness and derandomization

Keywords and phrases Pseudorandom generators, Constant-depth formulas, Explicit constructions

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.16

Funding *Dean Doron*: NSF Grant CCF-1705028.
Pooya Hatami: Simons Investigator Award (#409864, David Zuckerman).
William M. Hoza: NSF GRFP under Grant DGE-1610403 and a Harrington Fellowship from UT Austin.

Acknowledgements We thank David Zuckerman for very helpful discussions. The first author would also like to thank Gil Cohen, Chin Ho Lee and Amnon Ta-Shma for insightful conversations about the Forbes-Kelley result [17].

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka; Article No. 16; pp. 16:1–16:34

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

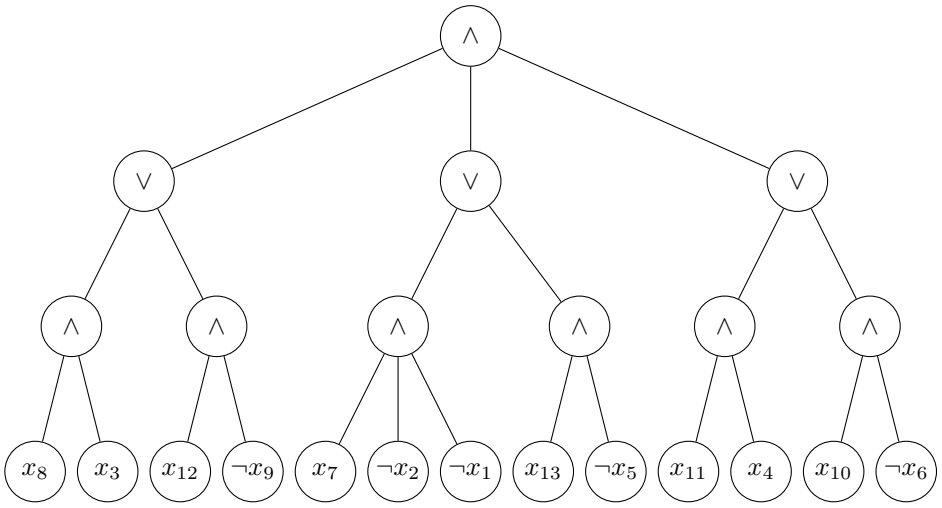


Figure 1 A depth-3 read-once $\mathbf{AC}^0$ formula on $n = 13$ bits.

1 Introduction

In complexity theory and algorithm design, randomness is a valuable yet scarce resource. A powerful, black-box method for reducing the randomness used by a computationally bounded process is to construct a *pseudorandom generator* (PRG). A PRG for a class of tests $\mathcal{C}$ is an algorithm that stretches a short truly random seed to a long n-bit string that "fools" $\mathcal{C}$, i.e., any test $f \in \mathcal{C}$ behaves the same on the output of the PRG as it does on a truly random string, up to some error ε.

Ideally, one would like to construct explicit unconditional PRGs with short seed length that fool powerful classes such as general polynomial-time algorithms. Unfortunately, constructing such general-purpose PRGs requires proving circuit lower bounds that seem to be far beyond the reach of state of the art techniques.

On the bright side, there has been a lot of success designing PRGs for more restricted classes. The two most intensely studied classes are read-once small-space algorithms and constant-depth circuits. In this work, we study *constant-depth read-once formulas* with unbounded fan-in over the basis $\{\wedge, \vee, \neg\}$ (Figure 1). This class is the read-once version of $\mathbf{AC}^0$. We construct an explicit PRG for this class with seed length $\widetilde{O}(\log(n/\varepsilon))$, which is optimal up to $\log\log$ factors.[1]

▶ **Theorem 1.** *For any positive integers n, d and for any $\varepsilon > 0$, there is an explicit ε-PRG for depth-d read-once $\mathbf{AC}^0$ formulas over n variables with seed length*

$$\log(n/\varepsilon) \cdot O(d \log\log(n/\varepsilon))^{2d+2}.$$

[1] A standard probabilistic argument shows the existence of a PRG with seed length $O(\log(n/\varepsilon))$. One can show a matching $\Omega(\log(n/\varepsilon))$ lower bound even for the depth-2 case.

1.1 Motivation and related work

Derandomizing Small-Space Algorithms

We are motivated by the **L** vs. **BPL** problem – namely whether every bounded-error probabilistic algorithm can be fully derandomized with only a constant factor space blowup. The way a log-space algorithm acts on its random bits can be modeled by a polynomial-width *read-once branching program* (ROBP). A natural approach to the **L** vs. **BPL** problem is thus coming up with a PRG for such ROBPs with seed length $O(\log n)$. Seminal work of Nisan gave a PRG with seed length $O(\log^2 n)$ for this model [35]. To this day, no better PRG is known even for ROBPs where the width is a large *constant*, though better generators are known in special cases [40, 14, 28, 41, 7, 22, 4, 10, 32].

Surprisingly, the study of fooling constant-width ROBPs has so far been closely entangled with the study of fooling read-once $\mathbf{AC}^0$. A depth-d read-once $\mathbf{AC}^0$ formula can be computed by a width-$(d+1)$ ROBP, possibly after reordering the inputs [13]. In the other direction, Gopalan et al. constructed a near-optimal PRG for read-once CNFs, and then used that PRG to construct a near-optimal hitting set generator for width-3 ROBPs [22]. Very recently, following the paradigm of Gopalan et al. [22], Meka, Reingold, and Tal gave a PRG for general width-3 ROBPs with near-optimal seed length when ε is constant [32].

Meanwhile, for any constant d, Chen, Steinke and Vadhan constructed a PRG for depth-d read-once $\mathbf{AC}^0$ formulas with seed length $\widetilde{O}(\log^{d+1} n)$ [13].[2] They obtained this PRG by proving new Fourier tail bounds for such formulas. Subsequently, Chattopadhyay et al. proved similar tail bounds for the stronger class of general width-$(d+1)$ ROBPs with arbitrarily ordered inputs; they used these tail bounds to construct a PRG with similar seed length for that model [11].

In a recent breakthrough, Forbes and Kelley gave an elegant construction of a PRG for ROBPs with arbitrarily ordered inputs [17]. In the polynomial-width case, their PRG has seed length $O(\log^3 n)$. For width-$(d+1)$ ROBPs when d is small, their PRG has seed length $\widetilde{O}(d \log^2 n)$; prior to the present work, this was also the best PRG for read-once $\mathbf{AC}^0$. Note that Theorem 1 improves on the Forbes-Kelley PRG [17] even for non-constant d, e.g., if $d = 0.2 \log\log n / \log\log\log n$ and $\varepsilon = 1/\operatorname{poly}(n)$.

Given the recent trend of connections between PRGs for ROBPs and PRGs for read-once $\mathbf{AC}^0$, we hope that our result will serve as a stepping stone toward optimal PRGs for general constant-width ROBPs.

Fooling General Constant-Depth Circuits

Ajtai and Wigderson were the first to consider the problem of fooling general $\mathbf{AC}^0$ circuits, and in their pioneering work they achieved seed length $O(n^\gamma)$ for any constant $\gamma > 0$ [1]. A long line of research has worked on improving this seed length [34, 29, 31, 3, 36, 6, 15, 21, 43, 42, 24, 38]. Today, for constant error, the best PRG for depth-d $\mathbf{AC}^0$ circuits known, by Tal, has seed length $\widetilde{O}(\log^{d+2} n)$ [42]. When ε is small, the best PRG is a very recent construction by Servedio and Tan [38], which achieves seed length $O(\log^{d+C} n \log(1/\varepsilon))$ for some unspecified absolute constant C.

[2] Note that Nisan's generator [35] is not guaranteed to fool read-once $\mathbf{AC}^0$ formulas because of the issue of variable ordering [5].

Fooling More General Read-Once Formulas

Bogdanov, Papakonstantinou, and Wan gave the first PRG for *unbounded-depth* read-once formulas [5]. Their PRG has seed length $(1 - \Omega(1))n$. More generally, their PRG fools formulas over an *arbitrary basis*, provided the fan-in is at most $O(n/\log n)$. For the case that the basis is $\{\wedge, \vee, \neg\}$, Impagliazzo, Meka, and Zuckerman gave an improved PRG for unbounded-depth read-once formulas with seed length $O(n^{0.2342})$ [26]. This was further improved by Forbes and Kelley [17]; their recent PRG with seed length $O(\log^3 n)$ fools unbounded-depth read-once formulas over an arbitrary basis with constant fan-in.

In another direction, Gavinsky, Lovett, and Srinivasan gave a PRG for constant-depth read-once formulas over the basis $\{\wedge, \vee, \neg, \mathrm{MOD}_m\}$, i.e., read-once $\mathbf{ACC}^0$ [19]. When the modulus m and the error ε are constant, their PRG has seed length $2^{O(d^2)} \cdot \log^{O(d)} n$; this result is also subsumed by the recent work of Forbes and Kelley [17]. As a reminder, in the present work, we focus on constant-depth read-once formulas over the $\{\wedge, \vee, \neg\}$ basis with unbounded fan-in.

Fooling Read-k Depth-2 Formulas

De et al. gave a PRG for read-once CNFs with seed length $O(\log n \log(1/\varepsilon))$ [15]; this result can also be deduced from earlier work by Chari, Rohatgi, and Srinivasan [8]. As mentioned previously, Gopalan et al. gave a PRG for read-once CNFs with seed length $\widetilde{O}(\log(n/\varepsilon))$ [22]. Meanwhile, Klivans, Lee, and Wan constructed a PRG that fools read-k CNFs even for small $k > 1$ [27]. Building on their work, Servedio and Tan recently gave an improved PRG for read-k CNFs [39]; if the size of the CNF is $\mathrm{poly}(n)$, their PRG has seed length $\log n \cdot \mathrm{poly}(k, \log(1/\varepsilon))$.

1.2 Overview of our Construction and Analysis

1.2.1 The Ajtai-Wigderson Approach

Our PRG follows the paradigm pioneered by Ajtai and Wigderson [1] and further developed by Gopalan et al. [22]. We begin by briefly explaining this general approach for constructing PRGs. Ultimately, to fool a test f, we want to pseudorandomly assign values to its inputs in such a way that f accepts or rejects with approximately the same probability as it would under a truly random input. As a first step, we pseudorandomly choose a *partial* assignment to f. Equivalently, we pseudorandomly choose a *restriction* $X \in \{0, 1, \star\}^n$, where $X_i = \star$ indicates that the variable X_i is still unset.

We need our pseudorandom distribution over restrictions to satisfy two key properties. The first property is that the restriction should approximately *preserve the expectation* of the function, i.e., in expectation over X, the restricted function $f|_X$ should have approximately the same bias as f itself. This feature ensures that after sampling the pseudorandom restriction X, our remaining task is simply to fool the restricted function $f|_X$.

The second property is that the restriction should *simplify* f, i.e., with high probability[3] over the pseudorandom restriction X, the restricted function $f|_X$ should in some sense be simpler than f itself. The purpose of this feature is that simplifying f should make it easier to fool, perhaps using a PRG from prior work. We shall now give a brief exposition of how we achieve these two properties in our work.

[3] In principle, it would actually suffice for f to merely simplify *in expectation* over X.

1.2.2 Preserving the Expectation Using the Work of Forbes and Kelley

Building on several prior works [37, 23, 11], Forbes and Kelley constructed a very simple pseudorandom distribution over restrictions that approximately preserves the expectation of any constant-width ROBP [17], hence any read-once $\mathbf{AC}^0$ formula. In the Forbes-Kelley distribution, the locations of the $\star$-s are chosen almost k-wise independently, and the non-$\star$ coordinates are filled in using a small-bias space. Each coordinate is $\star$ with probability roughly $\frac{1}{2}$, and the distribution can be sampled using $\widetilde{O}(\log(n/\varepsilon))$ truly random bits.

In our setting, we will design our restriction in such a way that the distribution of $\star$ locations is almost k-wise independent and the distribution of bits in the non-$\star$ coordinates has small bias, in addition to other properties we also need. That way, to argue that the expectation of the formula is preserved under our pseudorandom restriction, we can simply appeal to the Forbes-Kelley result [17].

1.2.3 Simplifying the Formula *Given* a PRG

The remaining challenge is to ensure that our pseudorandom restriction *simplifies* $\mathbf{AC}^0$ formulas. In the work of Forbes and Kelley [17], the measure of complexity was simply the number of remaining unset variables. That is, Forbes and Kelley argued that after applying $O(\log n)$ independent pseudorandom restrictions, with high probability, all variables are set, and hence there is nothing left to fool [17].[4] This gives them an overall seed length of $\widetilde{O}(\log(n/\varepsilon)\log n)$.

In this work, to achieve seed length $\widetilde{O}(\log(n/\varepsilon))$, we use a more sophisticated pseudorandom restriction and subtler measures of complexity. That way, we can argue that after applying just $\mathrm{poly}(\log\log(n/\varepsilon))$ independent restrictions, the formula has simplified enough that it can be fooled by a prior PRG.

Several "pseudorandom switching lemmas" are already known for $\mathbf{AC}^0$ [1, 43, 20, 38], but we were not able to use these lemmas for our result. Instead, the starting point for our approach to simplification is the work of Chen, Steinke, and Vadhan [13]. Chen et al. analyzed the effect of *truly* random restrictions on read-once $\mathbf{AC}^0$ formulas [13]. They showed that with high probability, a truly random restriction dramatically simplifies the formula in the sense that every node in the restricted formula has very few remaining children[5] [13]. Chen et al. mentioned that they would have liked to show that the same is true under pseudorandom restrictions – this would have improved the parameters of their main result – but they were not able to prove such a statement [13].

A key insight in our work is that roughly speaking, the predicate that some node is still alive after a random restriction X can be computed by *another read-once* $\mathbf{AC}^0$ *formula* whose inputs are the bits encoding X. Therefore, to pseudorandomly sample a restriction X that kills off each node with approximately the right probability, it suffices to select the bits encoding X using a PRG for read-once $\mathbf{AC}^0$. (Gavinsky, Lovett, and Srinivasan used a similar idea to fool read-once $\mathbf{ACC}^0$ [19].)

1.2.4 Obtaining the Necessary PRG Through Recursion

It may strike the reader that we have reached a "chicken or egg" problem: we can simplify formulas *given* a PRG for read-once $\mathbf{AC}^0$, but the whole reason we are interested in simplifying formulas is to *design* an improved PRG for read-once $\mathbf{AC}^0$! We resolve this difficulty by

[4] Actually, to get the best dependence on ε, Forbes and Kelley stop applying restrictions once the number of remaining variables drops below $O(\log n)$.

[5] A technicality is that this is only true "up to sandwiching."

recursing on the depth of the formula we wish to fool. That is, we assume we already have a PRG G_d that fools depth-d read-once $\mathbf{AC}^0$ formulas, and we use G_d to sample pseudorandom restrictions that simplify depth-$(d+1)$ read-once $\mathbf{AC}^0$ formulas. (This is similar to the approach of Gavinsky et al. [19].) Making this idea work requires overcoming several technical challenges.

In more detail, consider a collection of nodes $\{\phi_1, \ldots, \phi_k\}$ that form subformulas of depth $d' \leq d-1$. Roughly speaking, we show how to test the predicate that they are all still alive by a formula T of depth $d'+1 \leq d$.[6] The recursive generator G_d fools T, so under our pseudorandom restriction, the probability that $\phi_1, \ldots, \phi_k$ all remain alive is roughly what it would be under a truly random restriction.

Unfortunately, to ensure that the Forbes-Kelley analysis applies to our scenario, we are forced to design our pseudorandom restriction so that each coordinate is $\star$ with constant probability. The pseudorandom restriction has a similar effect as a truly random restriction with the same $\star$-probability, but that is not good enough. The analysis of truly random restrictions by Chen et al. only applies to the case that the $\star$-probability is $1/\operatorname{polylog}(n/\varepsilon)$ [13].

Roughly speaking, we overcome this difficulty using a kind of hybrid argument. A truly random restriction with $\star$-probability $1/\operatorname{polylog}(n/\varepsilon)$ is equivalent to the composition of t independent truly random restrictions, each with constant $\star$-probability, where $t = O(\log\log(n/\varepsilon))$. We show that for the purpose of simplification, a composition of t independent copies of our pseudorandom restriction is almost as good. Each individual step of this hybrid argument relies on the fact that G_d fools a formula closely related to the formula T mentioned earlier.

By applying an argument due to Gopalan et al. [22], we relate the condition that a collection of gates all remain alive to the number of remaining children of each node. Altogether, these arguments show that after applying $\operatorname{poly}(\log\log(n/\varepsilon))$ independent copies of our pseudorandom restriction, every gate *other than the root* has at most $\operatorname{polylog}(n)$ remaining children.[7] (We are not able to establish such a bound for the root gate, because its children form subformulas of depth $d' = d$.) Fortunately, this condition is strong enough that the restricted formula is fooled by a recent PRG by Meka, Reingold, and Tal [32]. We use the MRT PRG [32] as the last step in our construction.

1.3 Extension to Read-Once $\mathrm{AC}^0[\oplus]$ with a Few Parity Gates

$\mathbf{AC}^0$ is admittedly a fairly weak circuit class. The parity function is the most famous example of a function that cannot be computed in $\mathbf{AC}^0$ (e.g., [18, 25]). Having shown how to fool read-once $\mathbf{AC}^0$, the natural next problem is to fool read-once $\mathbf{AC}^0[\oplus]$, i.e., constant-depth read-once formulas over the basis $\{\oplus, \wedge, \vee, \neg\}$ with unbounded fan-in. Read-once $\mathbf{AC}^0[\oplus]$ can still be simulated by constant-width ROBPs (possibly after reordering the inputs), so fooling read-once $\mathbf{AC}^0[\oplus]$ would be another step on the long road to derandomizing $\mathbf{BPL}$. The best prior PRG for this model is once again Forbes and Kelley's PRG with seed length $\widetilde{O}(\log^2 n + \log n \log(1/\varepsilon))$ [17].

Fooling general (not necessarily read-once) $\mathbf{AC}^0[\oplus]$ circuits is a notoriously difficult problem in unconditional pseudorandomness. Currently, the best seed length is only slightly less than n [16].

[6] See Claim 10 for the precise statement.
[7] Again, this is only true up to sandwiching.

There has been more success fooling $\mathbf{AC}^0[\oplus]$ circuits under the assumption that the circuit only has a few parity gates [44, 9, 30]. In the same spirit, we show that our PRG for read-once $\mathbf{AC}^0$ formulas also fools read-once $\mathbf{AC}^0[\oplus]$ formulas with a bounded number of parity gates. We achieve seed length $\widetilde{O}(t + \log(n/\varepsilon))$, where t is the number of parity gates:

▶ **Theorem 2.** *For any positive integers n, d, t and for any $\varepsilon > 0$, there is an explicit ε-PRG for depth-d read-once $\mathbf{AC}^0[\oplus]$ formulas with at most t parity gates with seed length*

$$(td + \log(n/\varepsilon)) \cdot O(d \log\log(n/\varepsilon) + d\log(td))^{2d+2}.$$

At a very high level, this extension to $\mathbf{AC}^0[\oplus]$ is possible because the MRT PRG [32] was already designed for parities of small ROBPs. However, suitably extending the analysis of truly random restrictions by Chen et al. [13] to the case of $\mathbf{AC}^0[\oplus]$ is nontrivial. We defer further discussion to Section 9.

2 Preliminaries

2.1 Pseudorandomness Primitives

Let U_n denote the uniform distribution over $\{0,1\}^n$. Suppose $\mathcal{C}$ is a class of functions $f\colon \{0,1\}^n \to \mathbb{R}$ and G is a distribution over $\{0,1\}^n$. We say that G *ε-fools* $\mathcal{C}$ if for every $f \in \mathcal{C}$,

$$|\mathbb{E}[f(G)] - \mathbb{E}[f(U_n)]| \leq \varepsilon.$$

As two special cases, a *δ-biased* distribution is one that δ-fools parity functions, and a *γ-almost k-wise independent* distribution is one that γ-fools Boolean k-juntas [33, 2]. An *ε-PRG* for $\mathcal{C}$ is a function $G\colon \{0,1\}^s \to \{0,1\}^n$ such that $G(U_s)$ ε-fools $\mathcal{C}$. As a shorthand, we will write $\mathbb{E}[f]$ to denote $\mathbb{E}[f(U_n)]$.

2.2 Read-Once Formulas

An $\mathbf{AC}^0$ *formula* on $\{0,1\}^n$ is a rooted tree in which each internal node ("gate") is labeled with $\wedge$ or $\vee$ and each leaf is labeled with a constant (0 or 1), a variable x_i, or its negation $\neg x_i$, where $i \in [n]$. Gates may have arbitrary fan-in. The formula computes a function $\phi\colon \{0,1\}^n \to \{0,1\}$ in the natural way. The *depth* of the formula is the length of the longest path from the output gate to a leaf. The formula is *read-once* if each variable x_i appears at most once. We make no assumptions about the order in which the variables appear. A *layered* $\mathbf{AC}^0$ formula is one in which the gates are arranged in alternating layers of $\wedge$ and $\vee$ gates. Any read-once $\mathbf{AC}^0$ formula can be simulated by a layered read-once $\mathbf{AC}^0$ formula of the same depth.

2.3 Random Restrictions

A *restriction* is a string $x \in \{0,1,\star\}^n$. We define an associative *composition* operation on $\{0,1,\star\}^n$ by

$$(x \circ x')_i = \begin{cases} x_i & \text{if } x_i \neq \star \\ x'_i & \text{if } x_i = \star. \end{cases}$$

Conceptually, $x \circ x'$ corresponds to first restricting according to x and then further restricting according to x'. As a special case, if $x' \in \{0,1\}^n$, then $x \circ x' \in \{0,1\}^n$ is the string obtained

by using x' to "fill in the $\star$ positions" of x. If $f\colon \{0,1\}^n \to \{0,1\}$ is a function and x is a restriction, we define the restricted function $(f|_x)\colon \{0,1\}^n \to \{0,1\}$ by

$$(f|_x)(x') = f(x \circ x').$$

We define R_n to be the distribution over $X \in \{0,1,\star\}^n$ in which the coordinates are independent, $\Pr[X_i = \star] = 1/2$, and $\Pr[X_i = 0] = \Pr[X_i = 1] = 1/4$. If H_1, H_2 are distributions over $\{0,1,\star\}^n$, we define $H_1 \circ H_2$ to be the distribution over $X \in \{0,1,\star\}^n$ obtained by drawing independent samples $X_1 \sim H_1, X_2 \sim H_2$ and composing them, $X = X_1 \circ X_2$. For a nonnegative integer s, we define

$$H^{\circ s} = \underbrace{H \circ H \circ \cdots \circ H}_{s \text{ times}}.$$

For example, $R_n^{\circ s}$ is a random restriction where each coordinate is $\star$ with probability 2^{-s} and the non-$\star$ positions are uniform random bits.

A restriction can be specified by two n-bit strings as follows. Define $\mathrm{Res}\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1,\star\}^n$ by

$$\mathrm{Res}(y,z)_i = \begin{cases} \star & \text{if } y_i = 1 \\ z_i & \text{if } y_i = 0. \end{cases}$$

In words, y indicates which positions have $\star$, and z specifies the bits in the non-$\star$ positions. Observe that $\mathrm{Res}(U_{2n}) \sim R_n$.

3 Our PRG Construction

The construction of our generator is by induction on the depth of the formula we wish to fool. For the base case of depth-2 formulas, we use the PRG by Gopalan et al. for read-once CNFs and DNFs [22]. For the inductive step, let $d \geq 2$ be arbitrary, let G_d be a random variable over $\{0,1\}^n$ that α-fools depth-d read-once $\mathbf{AC}^0$ formulas, and let $\varepsilon > 0$ be arbitrary. We will show how to ε-fool depth-$(d+1)$ formulas, assuming α is sufficiently small.

Step 1: XORing with Small-Bias and Almost k-wise Independence

Let G_d' be an independent copy of G_d. Sample T from a γ-almost k-wise independent distribution over $\{0,1\}^n$, and sample D from a δ-biased distribution over $\{0,1\}^n$, where the parameters γ, k, δ will be specified later. Define

$$\overline{G}_d = (G_d \oplus T, G_d' \oplus D) \in \{0,1\}^n \times \{0,1\}^n.$$

Step 2: Assigning Most Inputs Using $\overline{G}_d$

Define a pseudorandom restriction $H_d \in \{0,1,\star\}^n$ by

$$H_d = \mathrm{Res}(\overline{G}_d).$$

Since $\mathrm{Res}(U_{2n}) \sim R_n$, each coordinate of H_d is $\star$ with probability roughly $1/2$. For a parameter

$$s = O((d \log\log(n/\varepsilon)) \cdot \log\log n),$$

we will restrict according to $H_d^{\circ s}$, i.e., we will compose s independent copies of the restriction H_d.

Step 3: Assigning Remaining Inputs Using the MRT PRG

We rely on a PRG by Meka, Reingold, and Tal for XORs of short ROBPs [32]; we will discuss this in more detail in Section 7. Sample $G_{\mathrm{MRT}} \in \{0,1\}^n$ using this PRG. Our final PRG for depth-$(d+1)$ read-once $\mathbf{AC}^0$ is defined by

$$G_{d+1} = H_d^{\circ s} \circ G_{\mathrm{MRT}},$$

i.e., we use G_{MRT} to assign bits to all remaining $\star$-positions after restricting according to $H_d^{\circ s}$.

4 Pseudorandom Restrictions Preserve Expectation

Toward proving the correctness of our PRG, in this section, we will show that restricting a depth-$(d+1)$ formula using the distribution H_d approximately preserves the expectation of the formula.

The following lemma proved by Forbes and Kelley shows that bounded-width ROBPs behave nicely under pseudorandom restrictions that are defined by small biased distributions and almost k-wise independence. In the lemma, $\mathcal{L}(n,w;k)$ is defined to be the maximum of $\sum_{i=1}^{k}\sum_{S\subseteq[n],|S|=k}|\widehat{f}(S)|$ over all width-w ROBPs f, where $\widehat{f}(S)$ denotes the Fourier coefficient of f at S.

▶ **Lemma 3** (Lemma 7.2 from [17], rephrased). *Let T and D be independent random variables over $\{0,1\}^n$, which are sampled respectively from a γ-almost k-wise independent distribution and a δ-biased distribution. Let $f\colon\{0,1\}^n\to\{0,1\}$ be a width-w arbitrarily-ordered ROBP. Then,*

$$\left|\mathop{\mathbb{E}}_{U\sim U_n}[f(U)] - \mathop{\mathbb{E}}_{\substack{T,D\\V\sim U_n}}\left[f|_{\mathrm{Res}(T,D)}(V)\right]\right| \le \left(\sqrt{\delta}\cdot\mathcal{L}(n,w;k) + \left(\frac{1}{2}\right)^{k/2} + \sqrt{\gamma}\right)\cdot nw.$$

We are mainly interested in fooling $\mathbf{AC}^0$ formulas, but for the analysis, it will be helpful to consider NAND *formulas*, i.e., formulas in which each internal node is a NAND gate instead of an $\wedge$ gate or an $\vee$ gate. In Section 8, we will explain why it suffices to reason about NAND formulas.

Recall from Section 3 that $\overline{G}_d = (G_d \oplus T, G'_d \oplus D)$, where G_d and G'_d are independent random variables over $\{0,1\}^n$ that α-fool depth-d read-once formulas, T is sampled from a γ-almost k-wise independent distribution over $\{0,1\}^n$, and D is sampled from a δ-biased distribution over $\{0,1\}^n$. We will use the following simple application of the above lemma to our pseudorandom restriction $H_d = \mathrm{Res}(\overline{G}_d)$. Looking ahead, we will eventually choose $\varepsilon_0 = \varepsilon/\operatorname{poly}(\log\log(n/\varepsilon))$.

▶ **Lemma 4.** *There exist constants $c_1, c_2, c_3 > 0$, such that for all positive integers n, d, for every $\varepsilon_0 > 0$, if we set*

$$k = c_1\log(nd/\varepsilon_0), \quad \delta = \varepsilon_0\cdot\left(\frac{c_2}{\log n}\right)^{-k(d+2)} \quad \text{and} \quad \gamma = \frac{c_3\varepsilon_0}{nd},$$

then H_d as defined above satisfies the following. For every depth-$(d+1)$ read-once NAND *formula $\phi\colon\{0,1\}^n\to\{0,1\}$,*

$$\left|\mathop{\mathbb{E}}_{U\sim U_n}[\phi(U)] - \mathop{\mathbb{E}}_{H_d,V\sim U_n}[\phi|_{H_d}(V)]\right| \le \varepsilon_0.$$

Proof. We start by noting that $G_d \oplus T$ and $G'_d \oplus D$ are independent, $G_d \oplus T$ is γ-almost k-wise independent, and $G'_d \oplus D$ is δ-biased. This is due to the fact that linear tests and k-juntas are closed under shifts.

The lemma is then an immediate corollary of Lemma 3, because every depth-$(d+1)$ read-once NAND formula can be computed by a width $d+2$ read-once branching program [13], and $\mathcal{L}(n, d+2; k)$ is bounded by $O(\log n)^{k(d+2)}$ [11]. Thus

$$\left| \mathop{\mathbb{E}}_{U \sim U_n} [\phi(U)] - \mathop{\mathbb{E}}_{H_d, V \sim U_n} [\phi|_{H_d}(V)] \right| \leq \left(\sqrt{\delta} \cdot O(\log n)^{k(d+2)} + \left(\frac{1}{2}\right)^{k/2} + \sqrt{\gamma} \right) \cdot n(d+2),$$

and it is easy to check that there are constants c_1, c_2, c_3 such that the right hand side is bounded by ε_0 for a choice of δ, γ, k as in the statement of the lemma. ◀

We get the following corollary about repeated applications of H_d immediately since depth-$(d+1)$ read-once formulas are closed under restrictions.

▶ **Corollary 5.** *Let ϕ be a depth-$(d+1)$ read-once* NAND *formula over n variables. Let δ, k, γ be as in Lemma 4. Then, for every integer $t \geq 1$,*

$$\left| \mathop{\mathbb{E}}_{U \sim U_n} [\phi(U)] - \mathop{\mathbb{E}}_{H_d^{\circ t}, V \sim U_n} \left[\phi|_{H_d^{\circ t}}(V) \right] \right| \leq \varepsilon_0 t.$$

5 Pseudorandom Restrictions Simplify Read-Once Formulas

In this section, we derandomize the analysis of Chen et al. [13] and show that our pseudorandom restriction generator $H_d^{\circ t}$ simplifies depth-$(d+1)$ formulas, as we discussed in Section 1.2. We first introduce our progress measure.

▶ **Definition 6.** *Given a read-once* NAND *formula ϕ, we let $\Delta(\phi)$ be the maximum fan-in of any gate in ϕ that is not the root.*

Our goal is to show that when X is sampled from $H_d^{\circ t}$ then a read-once formula ϕ is simplified in the sense that $\Delta(\phi|_X)$ is roughly $\sqrt{\Delta(\phi)}$, with high probability. We will show that $t = O(d \log\log(n/\varepsilon))$ is sufficient. Our analysis will closely follow the analysis by Chen et al. [13] for truly random restrictions.

5.1 Truly Random Restrictions Simplify Depth-$(d-1)$ Formulas

Chen, Steinke and Vadhan proved that biased read-once formulas collapse to a constant after a random restriction, with high probability [13]. Looking ahead, we will eventually set $\theta = (\varepsilon/n)^{O(1)}$.

▶ **Lemma 7** ([13], Lemma A.3)**.** *Let φ be a depth-d read-once* NAND *formula over n variables such that either $\mathbb{E}[\neg\varphi] \leq \rho$ or $\mathbb{E}[\varphi] \leq \rho$ for some $\rho \leq \frac{1}{2}$. Then, for every $\theta \in (0, \frac{2}{n})$ and $p \leq \frac{1}{(9\log(2 \cdot 4^d n/\theta))^d}$ it holds that*

$$\Pr_{X \sim R_n^{\circ \lceil \log p^{-1} \rceil}} [\varphi|_X \text{ is not a constant}] \leq 2p \cdot \rho \cdot (9\log(2 \cdot 4^d n/\theta))^d + \theta.$$

We use Lemma 7 to prove the following variation; note that this lemma considers the case of several read-once formulas and analyzes the probability of collapsing to 1 instead of collapsing to any constant.

▶ **Lemma 8.** *Let $\Phi = \{\phi_1, \ldots, \phi_k\}$ be a set of read-once* NAND *formulas over n variables, each of depth $d \leq \log n$ and over disjoint subsets of n variables. Further, assume that for every $i \in [k]$, $\mathbb{E}[\neg\phi_i] \leq \rho$ for some $\rho \leq \frac{1}{2}$. Then, there exists a constant c such that for every $\theta \in (0, \frac{2}{n})$ and integer $t \geq cd \log\log(n/\theta)$,*

$$\Pr_{X \sim R_n^{\circ t}}[\forall \phi \in \Phi, \phi|_X \not\equiv 1] \leq (2\rho + \theta)^k.$$

Proof. Consider some $\phi \in \Phi$ and let t be the smallest integer such that

$$2^{-t} \leq \frac{1}{2(9\log(2 \cdot 4^d n/\theta))^d},$$

and indeed $t = c\,(d \log\log(n/\theta) + d\log d)$ for some universal constant c. By Lemma 7,

$$\Pr_{X \sim R_n^{\circ t}}[\phi|_X \text{ is not a constant}] \leq \rho + \theta.$$

Now,

$$\Pr_{X \sim R_n^{\circ t}}[\phi|_X \equiv 0] \leq \mathbb{E}[\neg\phi] \leq \rho,$$

so by the union bound

$$\Pr_{X \sim R_n^{\circ t}}[\phi|_X \not\equiv 1] \leq 2\rho + \theta.$$

The lemma follows by the fact that each formula in Φ is over distinct variables and the coordinates of $R_n^{\circ t}$ are independent. ◀

5.2 H_d Simplifies Depth-$(d-1)$ Formulas

Ultimately, we are interested in the simplification of depth-$(d+1)$ formulas with respect to the $\Delta(\cdot)$ measure of progress. However, in this subsection, our goal is to prove that our iterated pseudorandom restriction $H_d^{\circ t}$ simplifies depth-$(d-1)$ formulas just as well as truly random restrictions up to an additive error. In this subsection, the notion of simplification is the event in the statement of Lemma 8.

▶ **Lemma 9.** *Let $\Phi = \{\phi_1, \ldots, \phi_k\}$ be a set of read-once* NAND *formulas over n variables, each of depth $d-1$ and over disjoint subsets of n variables. Then, for every integer $t \geq 1$,*

$$\Pr_{X \sim H_d \circ R_n^{\circ(t-1)}}[\forall \phi \in \Phi, \phi|_X \not\equiv 1] \leq \Pr_{X \sim R_n^{\circ t}}[\forall \phi \in \Phi, \phi|_X \not\equiv 1] + 2\alpha,$$

where α is the error of the PRG for depth-d read-once formulas underlying H_d.

Proof. Fix some restriction $v \in \{0,1,\star\}^n$. (Think of v as some fixing of $R_n^{\circ(t-1)}$.) Let $T_v \colon \{0,1\}^{2n} \to \{0,1\}$ be the predicate indicating that with respect to v, the given initial restriction does a poor job of simplifying Φ. That is,

$$T_v(y,z) = 1 \iff \forall \phi \in \Phi,\ \phi|_{\mathrm{Res}(y,z) \circ v} \not\equiv 1.$$

▷ Claim 10. For every $d \geq 2$, T_v can be computed by a depth-d read-once $\mathbf{AC}^0$ formula.

Proof. We will prove, by induction on d, that for every $\phi \in \Phi$,

1. The test $\phi|_{\mathrm{Res}(y,z) \circ v} \not\equiv 1$ can be computed by a depth-d read-once $\mathbf{AC}^0$ formula with an $\wedge$ gate on top.

2. The test $\phi|_{\mathrm{Res}(y,z)\circ v} \not\equiv 0$ can be computed by a depth-d read-once $\mathbf{AC}^0$ formula with an $\vee$ gate on top.

The claim will then follow, as the "$\forall \phi \in \Phi$" part is simply an $\wedge$ over formulas with a top $\wedge$ gate and thus the two top layers can be collapsed to a single layer.

For $d = 2$, ϕ is of depth 1 and so is simply a NAND of variables or their negation, say of the literals $\ell_1, \ldots, \ell_m$. Now,

$$\mathrm{NAND}(\ell_1, \ldots, \ell_m) \not\equiv 1 \iff \bigwedge_{i \in [m]} (\ell_i \not\equiv 0),$$

and

$$\mathrm{NAND}(\ell_1, \ldots, \ell_m) \not\equiv 0 \iff \bigvee_{i \in [m]} (\ell_i \not\equiv 1).$$

For each $b \in \{0, 1\}$, let us express the condition $\ell_i \not\equiv b$ in terms of the inputs y and z to T_v.

- If ℓ_i is a variable x_i, then

  $$x_i \not\equiv b \iff ((y_i = 1) \wedge (v_i \not\equiv b)) \vee ((y_i = 0) \wedge (z_i = \overline{b})).$$

  Now, v is fixed, so either $v_i \not\equiv b$ is the constant 0, in which case the formula amounts to $(y_i = 0) \wedge (z_i = \overline{b})$, or it is the constant 1, in which case the formula amounts to $(y_i = 1) \vee (z_i = \overline{b})$. Either way, this is a depth-1 read-once formula in terms of the inputs y and z to T_v.
- If ℓ_i is the negation $\neg x_i$ of some variable, then

  $$\neg x_i \not\equiv b \iff ((y_i = 1) \wedge (v_i \not\equiv \overline{b})) \vee ((y_i = 0) \wedge (z_i = b))$$

  Again, by the same reasoning, the above is a depth-1 read-once formula, where the top gate is determined by the value of $v_i \not\equiv b$.

Thus, the predicate $\mathrm{NAND}(\ell_1, \ldots, \ell_m) \not\equiv 1$ can be tested by a depth-2 formula where the top gate is an $\wedge$, and the predicate $\mathrm{NAND}(\ell_1, \ldots, \ell_m) \not\equiv 0$ can be tested by a depth-2 formula where the top gate is an $\vee$.

Assume the claim holds for some $d \geq 2$ and let $\phi = \mathrm{NAND}(\varphi_1, \ldots, \varphi_m)$ be a read-once NAND formula of depth d, so each φ_i is a depth-$(d-1)$ read-once NAND formula. We already mentioned that

$$\mathrm{NAND}(\varphi_1, \ldots, \varphi_m) \not\equiv 1 \iff \bigwedge_{i \in [m]} (\varphi_i \not\equiv 0).$$

By the induction's hypothesis, the predicate $\varphi_i|_{\mathrm{Res}(y,z)\circ v} \not\equiv 0$ can be tested by a depth-d read-once $\mathbf{AC}^0$ formula with a top $\vee$ gate, so overall we get a depth-$(d+1)$ read-once $\mathbf{AC}^0$ formula with a top $\wedge$ gate. Similarly,

$$\mathrm{NAND}(\varphi_1, \ldots, \varphi_m) \not\equiv 0 \iff \bigvee_{i \in [m]} (\varphi_i \not\equiv 1).$$

Again, by our assumption, the predicate $\varphi_i|_{\mathrm{Res}(y,z)\circ v} \not\equiv 1$ can be tested by a depth-d read-once $\mathbf{AC}^0$ formula with a top $\wedge$ gate, so overall we get a depth-$(d+1)$ read-once $\mathbf{AC}^0$ formula with a top $\vee$ gate. ◁

Recall from Section 3 the distribution

$$\overline{G}_d = (G_d \oplus T, G'_d \oplus D).$$

We shall later show:

▷ Claim 11. $\overline{G}_d$ (2α)-fools depth-d read-once $\mathbf{AC}^0$ formulas over $\{0,1\}^{2n}$.

With the above claim in mind, and Claim 10, we are now ready to proceed with proving the lemma. We get that:

$$\Pr_{X\sim H_d}[\forall \phi\in\Phi, \phi|_{X\circ v}\not\equiv 1] = \Pr_{X\sim H_d}[T_v(X)=1] \le \Pr_{(Y,Z)\sim U_{2n}}[T_v(Y,Z)=1] + 2\alpha.$$

A uniform (Y,Z) corresponds to a truly random restriction, so

$$\Pr_{X\sim H_d}[\forall \phi\in\Phi, \phi|_{X\circ v}\not\equiv 1] \le \Pr_{X\sim R_n}[\forall \phi\in\Phi, \phi|_{X\circ v}\not\equiv 1] + 2\alpha.$$

As the above is true for every restriction v, obviously

$$\mathop{\mathbb{E}}_{V\sim R_n^{\circ(t-1)}}\left[\Pr_{X\sim H_d}[\forall \phi\in\Phi, \phi|_{X\circ V}\not\equiv 1]\right] \le \mathop{\mathbb{E}}_{V\sim R_n^{\circ(t-1)}}\left[\Pr_{X\sim R_n}[\forall \phi\in\Phi, \phi|_{X\circ V}\not\equiv 1]\right] + 2\alpha,$$

so

$$\mathop{\mathbb{E}}_{X\sim H_d}\left[\Pr_{V\sim R_n^{\circ(t-1)}}[\forall \phi\in\Phi, \phi|_{X\circ V}\not\equiv 1]\right] \le \Pr_{X\sim R_n^{\circ t}}[\forall \phi\in\Phi, \phi|_{X}\not\equiv 1] + 2\alpha,$$

which amounts to what we wanted to prove. All that is left is to prove Claim 11.

Proof of Claim 11. We start by noting that since depth-d read-once $\mathbf{AC}^0$ is closed under shifts, $G_d\oplus T$ and $G'_d\oplus D$ both α-fool depth-d read-once $\mathbf{AC}^0$.

We will next use the fact that depth-d read-once $\mathbf{AC}^0$ is closed under restrictions. Suppose $\phi\colon \{0,1\}^n\times\{0,1\}^n\to\{0,1\}$ is a depth-d read-once $\mathbf{AC}^0$ formula. We have

$$\begin{aligned}
&\left|\mathop{\mathbb{E}}_{U,V\sim U_n}[\phi(U,V)] - \mathop{\mathbb{E}}_{(X,Y)\sim\overline{G}_d}[\phi(X,Y)]\right| \\
&\qquad\le \left|\mathop{\mathbb{E}}_{V\sim U_n}\left[\mathop{\mathbb{E}}_{U\sim U_n}[\phi(U,V)] - \mathop{\mathbb{E}}_{X\sim G_d\oplus T}[\phi(X,V)]\right]\right| \\
&\qquad\qquad + \left|\mathop{\mathbb{E}}_{X\sim G_d\oplus T}\left[\mathop{\mathbb{E}}_{V\sim U_n}[\phi(X,V)] - \mathop{\mathbb{E}}_{Y\sim G'_d\oplus D}[\phi(X,Y)]\right]\right| \\
&\qquad\le \mathop{\mathbb{E}}_{V\sim U_n}\left|\mathop{\mathbb{E}}_{U\sim U_n}[\phi(U,V)] - \mathop{\mathbb{E}}_{X\sim G_d\oplus T}[\phi(X,V)]\right| \\
&\qquad\qquad + \mathop{\mathbb{E}}_{X\sim G_d\oplus T}\left|\mathop{\mathbb{E}}_{V\sim U_n}[\phi(X,V)] - \mathop{\mathbb{E}}_{Y\sim G'_d\oplus D}[\phi(X,Y)]\right| \\
&\qquad\le 2\alpha,
\end{aligned}$$

where we used the fact that $G_d\oplus T$ and $G'_d\oplus D$ are independent and α-fool the formulas $\phi(\cdot,v)$ and $\phi(x,\cdot)$ respectively. ◁

◀

Iterating H_d for t times, we get the following lemma. Roughly speaking, the proof is a hybrid argument of which Lemma 9 is a single step.

▶ **Lemma 12.** *Let $\Phi=\{\phi_1,\dots,\phi_k\}$ be a set of read-once* NAND *formulas over n variables, each of depth $d-1$ and over disjoint subsets of n variables. Then, for every integer $t\ge 1$,*

$$\Pr_{X\sim H_d^{\circ t}}[\forall \phi\in\Phi, \phi|_X\not\equiv 1] \le \Pr_{X\sim R_n^{\circ t}}[\forall \phi\in\Phi, \phi|_X\not\equiv 1] + 2t\alpha,$$

where α is the error of the PRG for depth-d read-once $\mathbf{AC}^0$ formulas underlying H_d.

Proof. We prove the lemma by induction on t. The case of $t = 0$ is trivial. Now, assume that

$$\Pr_{X\sim H_d^{\circ(t-1)}}[\forall\phi\in\Phi,\phi|_X\not\equiv 1]\le\Pr_{X\sim R_n^{\circ(t-1)}}[\forall\phi\in\Phi,\phi|_X\not\equiv 1]+2(t-1)\alpha.$$

Thus,

$$\begin{aligned}\Pr_{X\sim H_d^{\circ t}}[\forall\phi\in\Phi,\phi|_X\not\equiv 1] &= \mathop{\mathbb{E}}_{X_1\sim H_d}\left[\Pr_{X_2\sim H_d^{\circ(t-1)}}[\forall\phi\in\Phi,\phi|_{X_1\circ X_2}\not\equiv 1]\right]\\ &= \mathop{\mathbb{E}}_{X_1\sim H_d}\left[\Pr_{X_2\sim H_d^{\circ(t-1)}}[\forall\phi\in\Phi,(\phi|_{X_1})\,|_{X_2}\not\equiv 1]\right]\\ &\le \mathop{\mathbb{E}}_{X_1\sim H_d}\left[\Pr_{X_2\sim R_n^{\circ(t-1)}}[\forall\phi\in\Phi,(\phi|_{X_1})\,|_{X_2}\not\equiv 1]\right]+2(t-1)\alpha\\ &\le \Pr_{X\sim R_n^{\circ t}}[\forall\phi\in\Phi,\phi|_X\not\equiv 1]+2t\alpha.\end{aligned}$$

The third transition used the induction's hypothesis and the last one is due to Lemma 9. ◀

Combining Lemma 12 with Lemma 8 we immediately get the following corollary.

▶ **Corollary 13.** *Let $\Phi=\{\phi_1,\ldots,\phi_k\}$ be a set of* NAND *read-once formulas over n variables, each of depth $d-1$ and over disjoint subsets of n variables. Further, assume that $d\le\log n$ and that for every $i\in[k]$, $\mathbb{E}[\neg\phi_i]\le\rho$ for some $\rho\le\frac{1}{2}$. Then, there exists a constant c such that for every $\theta\in(0,\frac{2}{n})$ and integer $t\ge cd\log\log(n/\theta)$,*

$$\Pr_{X\sim H_d^{\circ t}}[\forall\phi\in\Phi,\phi|_X\not\equiv 1]\le(2\rho+\theta)^k+2t\alpha,$$

where α is the error of the PRG for depth-d read-once $\mathbf{AC}^0$ formulas underlying H_d.

5.3 $H_d^{\circ t}$ Simplifies Depth-$(d+1)$ Formulas

We are now ready to prove our main result for this section.

▶ **Lemma 14.** *Let ϕ be a depth-$(d+1)$ read-once* NAND *formula over n variables where $d\le\log n$. Let $\varepsilon_0>0$ and let c be the constant guaranteed by Corollary 13. Further assume that $\theta\in(0,\frac{2}{n})$ is such that for every gate ψ in ϕ, possibly excluding the root, $\mathbb{E}[\neg\psi]\ge\theta$. Then, for every integer $t\ge cd\log\log(n/\theta)$ and every $\alpha\le\frac{\varepsilon_0^2}{8(dn)^2\sqrt{n}\log^2(1/\theta)t}$,*

$$\Pr_{X\sim H_d^{\circ t}}\left[\Delta(\phi|_X)\le 10\sqrt{\Delta(\phi)}\log^2(1/\theta)\right]\ge 1-\varepsilon_0,$$

where the PRG for depth-d read-once $\mathbf{AC}^0$ formulas underlying H_d is instantiated with error α.

Note that we assume here that every gate in ϕ has a non-negligible probability of rejecting, which may not always be the case. Following Chen et al. [13], in Section 6 we will get rid of that assumption by a sandwiching argument. The proof of Lemma 14 is based on an argument introduced by Gopalan et al. [22], later also used by Chen et al. [13].

Proof. Let ψ be any gate in ϕ other than the root, so ψ is a depth-d read-once NAND formula. We shall partition its children Ψ according to their rejection probability. Namely, for every integer $0 \leq i \leq \log(1/\theta) - 1$ define

$$\Psi_i = \left\{\varphi \in \Psi : 2^i\theta \leq \mathbb{E}[\neg\varphi] < 2^{i+1}\theta\right\}.$$

Note that if $\mathbb{E}[\neg\varphi] = 1$ then ψ is fixed to 1 so we can simply ignore it.

Let us fix some $0 \leq i \leq \log(1/\theta) - 1$ and consider the set of formulas Ψ_i. In hindsight, set the parameters

$$M = 5e\ln(1/\theta)\sqrt{\Delta(\phi)}$$

and

$$k = \left\lceil \frac{2}{\log \Delta(\phi)} \log\left(\frac{2dn\log(1/\theta)}{\varepsilon_0}\right)\right\rceil.$$

Write $\Psi_i = \{\varphi_1, \ldots, \varphi_w\}$. For every $j \in [w]$, let Y_j be the indicator for the event that φ_j is not identically 1 after a pseudorandom restriction, namely $\varphi_j|_X \not\equiv 1$. We wish to bound

$$\Pr\left[\sum_{j\in[w]} Y_j \geq M\right],$$

where the probability is taken over $X \sim H_d^{\circ t}$. Let

$$S_k(x_1, \ldots, x_w) = \sum_{I\subseteq[w], |I|=k} \prod_{i\in I} x_i$$

be the k-th elementary symmetric polynomial. Note that if $\sum_{j\in[w]} Y_j \geq M$ then $S_k(Y_1, \ldots, Y_w)$ is at least $\binom{M}{k}$, and so

$$\begin{aligned}\Pr\left[\sum_{j\in[w]} Y_j \geq M\right] &\leq \frac{1}{\binom{M}{k}}\mathbb{E}[S_k(Y_1, \ldots, Y_w)] \\ &\leq \left(\frac{k}{M}\right)^k \sum_{I\subseteq[w], |I|=k} \Pr\left[\forall j \in I, Y_j = 1\right].\end{aligned}$$

We know that $\mathbb{E}[\neg\varphi] \leq 2^{i+1}\theta$ and φ is a depth-$(d-1)$ NAND formula, so by Corollary 13 we get

$$\Pr\left[\sum_{j\in[w]} Y_j \geq M\right] \leq \left(\frac{k}{M}\right)^k \binom{w}{k}\left((2\cdot 2^{i+1}\theta + \theta)^k + 2t\alpha\right). \tag{1}$$

Now,

▷ Claim 15. It holds that $w \leq \frac{\ln(1/\theta)}{2^i\theta}$.

Proof. On the one hand,

$$\prod_{\varphi\in\Psi} \mathbb{E}[\varphi] = \mathbb{E}[\neg\psi] \geq \theta.$$

On the other hand,

$$\prod_{\varphi\in\Psi} \mathbb{E}[\varphi] \leq \prod_{\varphi\in\Psi_i} \mathbb{E}[\varphi] \leq (1 - 2^i\theta)^w \leq e^{-2^i w\theta}.$$

Combining the two gives the desired bound. ◁

Plugging in the above bound to Equation (1), we get

$$\begin{aligned}\Pr\left[\sum_{j\in[w]} Y_j \geq M\right] &\leq \left(\frac{k}{M}\right)^k \left(\frac{we}{k}\right)^k \left((2\cdot 2^{i+1}\theta+\theta)^k + 2t\alpha\right) \\ &\leq \left(\frac{ew\cdot(2^{i+2}\theta+\theta)}{M}\right)^k + 2\left(\frac{we}{M}\right)^k t\alpha \\ &\leq \left(\frac{5e\ln(1/\theta)}{M}\right)^k + 2\left(\frac{\Delta(\phi)e}{M}\right)^k t\alpha,\end{aligned}$$

where for the second summand we only used the trivial fact that $w \leq \Delta(\phi)$.

Plugging in M, we achieve

$$\Pr\left[\sum_{j\in[w]} Y_j \geq M\right] \leq \frac{1}{\Delta(\phi)^{k/2}} + 2(\Delta(\phi))^{k/2}\cdot t\alpha. \tag{2}$$

As $k \geq \frac{2}{\log\Delta(\phi)}\log\left(\frac{2dn\log(1/\theta)}{\varepsilon_0}\right)$ we have that the first summand of Equation (2) is at most $\frac{\varepsilon_0}{2dn\log(1/\theta)}$. Also, the bound on α implies

$$\frac{2dn\log(1/\theta)}{\varepsilon_0} \leq \frac{\varepsilon_0}{4dn\log(1/\theta)t\alpha}\cdot\frac{1}{\sqrt{\Delta(\phi)}}$$

so

$$k \leq \frac{2}{\log\Delta(\phi)}\log\left(\frac{2dn\log(1/\theta)}{\varepsilon_0}\right)+1 \leq \frac{2}{\log\Delta(\phi)}\log\left(\frac{\varepsilon_0}{4dn\log(1/\theta)t\alpha}\right)$$

and the second summand of Equation (2) is at most $\frac{\varepsilon_0}{2dn\log(1/\theta)}$ as well. Thus,

$$\Pr\left[\sum_{j\in[w]} Y_j \geq M\right] \leq \frac{\varepsilon_0}{dn\log(1/\theta)}.$$

Define $E_i = \sum_{j\in[w]} Y_j$. By union-bounding over $\Psi_0,\ldots,\Psi_{\log(1/\theta)-1}$ we get that

$$\Pr\left[\sum_{i=0}^{\log(1/\theta)-1} E_i \geq M\log(1/\theta)\right] \leq \sum_{i=0}^{\log(1/\theta)-1}\Pr[E_i] \leq \frac{\varepsilon_0}{dn}.$$

Another union bound over all possible ψ-s (at most dn of them) gives us the desired bound. ◀

6 Ensuring Noticeable Chance of Rejecting

In Section 5, we showed that $H^{\circ t}$ simplifies formulas with high probability *under the assumption* that every gate rejects with noticeable probability. In this section, following Chen, Steinke, and Vadhan [13], we will use a sandwiching argument to handle gates with negligible probability of rejecting. Our starting point is a helpful lemma implicit in the work of Chen et al. [13]:

▶ **Lemma 16** ([13]). *Suppose ϕ is a depth-d read-once* NAND *formula over n variables with $d \leq n$ and let $\varepsilon_0 > 0$. Define $\theta = \frac{\varepsilon_0^2}{4n^2}$. Then, there exist read-once* NAND *formulas ℓ_ϕ, u_ϕ with the following properties.*

1. $\ell_\phi \le \phi \le u_\phi$ *and* $\mathbb{E}[u_\phi - \ell_\phi] \le \varepsilon_0$.
2. *The underlying tree structure of* ℓ_ϕ *is a subgraph of the underlying tree structure of* ϕ*, and the underlying tree structure of* u_ϕ *is a subgraph of the underlying tree structure of* ϕ.
3. *Every non-constant gate* ψ *in either* ℓ_ϕ *or* u_ϕ *satisfies* $\mathbb{E}[\psi] \ge \theta$ *and* $\mathbb{E}[\neg\psi] \ge \theta$.

Since Chen, Steinke, and Vadhan did not state Lemma 16 exactly as we have stated it here, for completeness, we include a proof of Lemma 16 in Appendix A.

The sandwiching formulas in Lemma 16 satisfy the hypothesis of Lemma 14, so after restricting according to $H^{\circ t}$, they simplify in the sense that Δ goes down by roughly a square root. We would like to apply $H^{\circ t}$ again to simplify the formulas even further. Unfortunately, after the first application of $H^{\circ t}$, the restricted formulas might no longer satisfy the hypothesis of Lemma 14. Therefore, before applying $H^{\circ t}$ the second time, we must apply Lemma 16 again. We will continue in this manner, alternately applying $H^{\circ t}$ to simplify and applying Lemma 16 to eliminate gates with negligible probability of rejecting. In this way, we will prove the following lemma.

▶ **Lemma 17.** *Suppose* ϕ *is a depth-*$(d+1)$ *read-once* NAND *formula over* n *variables where* $d \le \log n$ *and let* $\varepsilon_0 > 0$*. Assume the parameters* $\alpha, k, \delta, \gamma$ *underlying* H_d *satisfy the hypotheses of Lemma 14 and Lemma 4. Let* θ *be the value in Lemma 16, let* t *be as in Lemma 14, let* $r = \lceil 3 \log\log n \rceil$*, and let* $s = rt$.

Sample independent restrictions $X_1, \dots, X_r \sim H_d^{\circ t}$*. For any such vector of restrictions* $\vec{X}$*, there exist depth-*$(d+1)$ *read-once* NAND *formulas* $\ell_{\phi,\vec{X}}, u_{\phi,\vec{X}}$ *with the following properties.*

1. *(Bounding.) For every sample* $\vec{X}$*,*

$$\ell_{\phi,\vec{X}} \le \phi|_{X_1 \circ \cdots \circ X_r} \le u_{\phi,\vec{X}}.$$

2. *(Sandwiching.) For* $U \sim U_n$ *independent of* $\vec{X}$*,*

$$\mathop{\mathbb{E}}_{\vec{X},U}\left[u_{\phi,\vec{X}}(U) - \ell_{\phi,\vec{X}}(U)\right] \le 3s\varepsilon_0.$$

3. *(Simplicity.) Let* $\Delta_0 = 40^4 \log^8(2n/\varepsilon_0)$*. Then,*

$$\Pr_{\vec{X}}\left[\Delta\left(\ell_{\phi,\vec{X}}\right) \le \Delta_0 \text{ and } \Delta\left(u_{\phi,\vec{X}}\right) \le \Delta_0\right] \ge 1 - 2r\varepsilon_0.$$

Toward proving Lemma 17, fix a depth-$(d+1)$ read-once NAND formula ϕ, define $X_0 = \star^n$, and define $\ell^{(0)}_{\vec{X}} = u^{(0)}_{\vec{X}} = \phi$. Then, for $i < r$, inductively define

$$\ell^{(i+1)}_{\vec{X}} = \ell_{(\ell^{(i)}_{\vec{X}}|_{X_i})}.$$

That is, $\ell^{(i+1)}_{\vec{X}}$ is the lower sandwiching formula when Lemma 16 is applied to $\ell^{(i)}_{\vec{X}}\big|_{X_i}$. Similarly, define

$$u^{(i+1)}_{\vec{X}} = u_{(u^{(i)}_{\vec{X}}|_{X_i})},$$

i.e., $u^{(i+1)}_{\vec{X}}$ is the upper sandwiching formula when Lemma 16 is applied to $u^{(i)}_{\vec{X}}\big|_{X_i}$. Finally, define

$$\ell_{\phi,\vec{X}} = \ell^{(r)}_{\vec{X}}\big|_{X_r}$$
$$u_{\phi,\vec{X}} = u^{(r)}_{\vec{X}}\big|_{X_r}.$$

Proof of Item 1 of Lemma 17. We show by induction on i that $\ell^{(i)}_{\vec{X}}|_{X_i} \leq \phi|_{X_1 \circ \cdots \circ X_i} \leq u^{(i)}_{\vec{X}}|_{X_i}$. In the base case $i = 0$, this is trivial. For the inductive step, we have

$$\begin{aligned}
\ell^{(i+1)}_{\vec{X}}|_{X_{i+1}} &\leq \left(\ell^{(i)}_{\phi}|_{X_i}\right)|_{X_{i+1}} && \text{By Item 1 of Lemma 16} \\
&\leq (\phi|_{X_1 \circ \cdots \circ X_i})|_{X_{i+1}} && \text{By the induction's hypothesis} \\
&= \phi|_{X_1 \circ \cdots \circ X_{i+1}}.
\end{aligned}$$

A completely analogous argument works for the upper bound as well. ◀

Proof of Item 2 of Lemma 17. We show by induction on i that

$$\mathop{\mathbb{E}}_{\vec{X},U}\left[u^{(i)}_{\vec{X}}|_{X_i}(U) - \ell^{(i)}_{\vec{X}}|_{X_i}(U)\right] \leq (2t+2)i\varepsilon_0. \tag{3}$$

In the base case $i = 0$, the statement is trivial. For the inductive step, we have

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\vec{X},U}\left[u^{(i+1)}_{\vec{X}}|_{X_{i+1}}(U) - \ell^{(i+1)}_{\vec{X}}|_{X_{i+1}}(U)\right] \\
&\quad\leq \mathop{\mathbb{E}}_{\vec{X},U}\left[u^{(i+1)}_{\vec{X}}(U) - \ell^{(i+1)}_{\vec{X}}(U)\right] + 2t\varepsilon_0 && \text{By Corollary 5} \\
&\quad\leq \mathop{\mathbb{E}}_{\vec{X},U}\left[u^{(i)}_{\vec{x}}|_{X_i}(U) + \ell^{(i)}_{\vec{X}}|_{X_i}(U)\right] + (2t+2)\varepsilon_0 && \text{By Item 1 of Lemma 16} \\
&\quad\leq (2t+2)(i-1)\varepsilon_0 + (2t+2)\varepsilon_0. && \text{By the induction's hypothesis}
\end{aligned}$$

Finally, Item 2 of Lemma 17 follows from Equation (3) by plugging-in $i = r$ and as $s = rt$. ◀

Proof of Item 3 of Lemma 17. By construction, for every $i \geq 1$, the formula $\ell^{(i)}_{\vec{X}}$ and the formula $u^{(i)}_{\vec{X}}$ both have the property that every gate ψ satisfies $\mathbb{E}[\neg\psi] \geq \theta$, where

$$\theta = \frac{\varepsilon_0^2}{4n^2}.$$

Furthermore, as the restrictions are independent, X_i is independent of $\left(\ell^{(i)}_{\vec{X}}, u^{(i)}_{\vec{X}}\right)$. Therefore, by Lemma 14,

$$\Pr_{\vec{X}}\left[\Delta\left(\ell^{(i)}_{\vec{X}}|_{X_i}\right) > 10\sqrt{\Delta\left(\ell^{(i)}_{\vec{X}}\right)} \cdot \log^2(1/\theta)\right] \leq \varepsilon_0,$$

and

$$\Pr_{\vec{X}}\left[\Delta\left(u^{(i)}_{\vec{X}}|_{X_i}\right) > 10\sqrt{\Delta\left(u^{(i)}_{\vec{X}}\right)} \cdot \log^2(1/\theta)\right] \leq \varepsilon_0.$$

By the union bound, we may assume that none of these bad events occur and accumulate an error of $2\varepsilon_0$ for every restriction. Based on this assumption, we now show by induction on i that

$$\Delta\left(\ell^{(i)}_{\vec{X}}|_{X_i}\right) \leq \max\left\{10^4 \log^8(1/\theta), n^{(3/4)^i}\right\}, \tag{4}$$

and

$$\Delta\left(u^{(i)}_{\vec{X}}|_{X_i}\right) \leq \max\left\{10^4 \log^8(1/\theta), n^{(3/4)^i}\right\}. \tag{5}$$

The base case $i = 0$ follows from the trivial bound $\Delta(\phi) \leq n$. Now the inductive step. We have

$$\begin{aligned}
\Delta\left(\ell_{\vec{X}}^{(i+1)}\big|_{X_{i+1}}\right) &\leq 10\sqrt{\Delta\left(\ell_{\vec{X}}^{(i+1)}\right)} \cdot \log^2(1/\theta) && \text{By our assumption} \\
&\leq 10\sqrt{\Delta\left(\ell_{\vec{X}}^{(i)}\big|_{x_i}\right)} \cdot \log^2(1/\theta) && \text{By Item 2 of Lemma 16} \\
&\leq 10\sqrt{\max\left\{10^4 \log^8(1/\theta), n^{(3/4)^i}\right\}} \cdot \log^2(1/\theta) && \text{By the induction's hypothesis}
\end{aligned}$$

Now we have two cases. First, suppose $n^{(3/4)^i} \leq 10^4 \log^8(1/\theta)$. Then the bound becomes

$$\begin{aligned}
\Delta\left(\ell_{\vec{X}}^{(i+1)}\right) &\leq 10\sqrt{10^4 \log^8(1/\theta)} \cdot \log^2(1/\theta) \\
&= 10^3 \log^6(1/\theta) \\
&\leq 10^4 \log^8(1/\theta),
\end{aligned}$$

completing the proof of Equation (4) in this case. Now, suppose instead that $10^4 \log^8(1/\theta) < n^{(3/4)^i}$. Then the bound becomes

$$\begin{aligned}
\Delta\left(\ell_{\vec{X}}^{(i+1)}\right) &\leq 10\sqrt{n^{(3/4)^i}} \cdot \log^2(1/\theta) \\
&\leq \sqrt{n^{(3/4)^i}} \cdot (n^{(3/4)^i})^{1/4} \\
&= n^{(3/4)^{i+1}},
\end{aligned}$$

once again completing the proof of Equation (4). The proof of Equation (5) is completely analogous and we omit it. Item 3 of Lemma 17 follows because by our choice of r, $n^{(3/4)^r} \leq 2$, and by the definition of θ,

$$10^4 \log^8(1/\theta) = 40^4 \log^8(2n/\varepsilon_0).$$ ◀

7 Fooling Formulas When Δ is Small

Recall from Section 3 that our pseudorandom distribution for depth-$(d+1)$ read-once formulas is

$$H_d^{\circ s} \circ G_{\mathrm{MRT}}.$$

So far, we have shown that up to sandwiching, applying $H_d^{\circ s}$ substantially simplifies the formula with high probability while approximately preserving its expectation (Lemma 17). It remains to show that G_{MRT} fools these simpler formulas. Meka, Reingold, and Tal studied the problem of fooling XORs of short ROBPs and achieved the following parameters.

▶ **Theorem 18** ([32]). *For any positive integers n, w, b and any $\varepsilon_0 > 0$ there is an explicit PRG that ε_0-fools all functions $f\colon \{0,1\}^n \to \{\pm 1\}$ of the form*

$$f(x) = \prod_{i=1}^{m} g_i(x),$$

where $g_1, \ldots, g_m : \{0,1\}^n \to \{\pm 1\}$ are defined over disjoint variable sets of size at most b and each g_i can be computed by an arbitrarily ordered width-w ROBP. The seed length of the PRG is

$$\log(n/\varepsilon_0) \cdot O(\log b + \log\log(n/\varepsilon_0))^{2w+2}.$$

It immediately follows that we can fool formulas when Δ is small with the following parameters.

▶ **Corollary 19.** *For any integers n, d, Δ_0 and any $\varepsilon_0 > 0$, there is an explicit distribution G_{MRT} that ε_0-fools depth-d read-once* NAND *formulas ϕ satisfying $\Delta(\phi) \le \Delta_0$ that can be sampled using*

$$\log(n/\varepsilon_0) \cdot O(d \log \Delta_0 + \log\log(n/\varepsilon_0))^{2d+2}$$

truly random bits.

Proof. Write $\phi = \mathrm{NAND}(\varphi_1, \ldots, \varphi_m)$. Then $\neg\phi = \wedge_{i=1}^m \varphi_i$. Applying the Fourier expansion of the m-input $\wedge$ function gives

$$\neg\phi = \sum_{S \subseteq [m]} \frac{(-1)^{|S|}}{2^m} \cdot \prod_{i \in S} (-1)^{\varphi_i}.$$

Since $\sum_S \left| \frac{(-1)^{|S|}}{2^m} \right| = 1$, it suffices to fool each function $\prod_{i \in S}(-1)^{\varphi_i}$ separately.

Since $\Delta(\phi) \le \Delta_0$, each φ_i depends on at most Δ_0^{d-1} variables. Since ϕ is read-once, the φ_i-s depend on disjoint sets of variables. Since each φ_i is a depth-$(d-1)$ read-once NAND formula, it can be computed by a width-d ROBP under some ordering of the variables [13]. Applying Theorem 18 completes the proof, since fooling ϕ is equivalent to fooling $\neg\phi$. ◀

8 Putting Everything Together: Proof of Theorem 1

To prove the correctness of our PRG, we first need to justify the fact that our analysis has so far focused on NAND formulas whereas our main result governs $\mathbf{AC}^0$ formulas, i.e., formulas over the $\{\wedge, \vee, \neg\}$ basis.

▶ **Lemma 20.** *For any layered read-once $\mathbf{AC}^0$ formula ϕ, either ϕ or $\neg\phi$ can be computed by a read-once* NAND *formula with the same underlying tree structure as ϕ.*

Proof. We proceed by induction on the depth d of ϕ to show that if the output gate of ϕ is $\vee$, then ϕ can be computed by a read-once NAND formula with the same underlying tree structure as ϕ. In the base case $d = 1$, we have $\phi = \vee_{i=1}^m \ell_i$, where each ℓ_i is a literal. Then we can also write

$$\phi = \mathrm{NAND}(\neg\ell_1, \ldots, \neg\ell_m).$$

Now, for the inductive step, assume $\phi = \vee_{i=1}^m \varphi_i$, where each φ_i is a depth-d read-once formula with output gate $\wedge$. Then once again,

$$\phi = \mathrm{NAND}(\neg\varphi_1, \ldots, \neg\varphi_m).$$

By moving $\neg$ gates downward, $\neg\varphi_i$ can be converted to a depth-d read-once formula with output gate $\vee$ without altering its underlying tree structure. Applying the induction's hypothesis completes the proof. Finally, the lemma follows, because if the output gate of ϕ is $\wedge$, then $\neg\phi$ can be computed by a read-once formula with the same underlying tree structure with output gate $\vee$. ◀

Conversely, any read-once NAND formula can be straightforwardly simulated by a layered read-once $\mathbf{AC}^0$ formula with the same underlying tree structure. We are now ready to complete the analysis of our PRG.

Proof of Theorem 1. Recall that our PRG is $G_{d+1} = H_d^{\circ s} \circ G_{\text{MRT}}$.

Parameters. Assume $d \leq \log\log(n/\varepsilon)$. (Otherwise, Theorem 1 follows already from the work of Forbes and Kelley [17].) Let c be the constant from Lemma 8. Let $r = \lceil 3 \log\log n \rceil$, and define

$$\varepsilon_0 = \frac{\varepsilon}{10r \cdot cd \log\log(n/\varepsilon)}.$$

Let $\theta = \frac{\varepsilon_0^2}{4n^2}$. Let $t = cd\lceil \log\log(n/\theta) \rceil$ (without loss of generality, take c to be an integer), and let $s = tr$. Let $\alpha = \varepsilon^4/n^3$; this is small enough to satisfy the hypothesis of Lemma 14. Let k, δ, γ be the values required by Lemma 4. Let Δ_0 be the value specified by Lemma 17.

Correctness. Let ϕ be a depth-$(d+1)$ read-once $\mathbf{AC}^0$ formula. We can straightforwardly make ϕ a *layered* read-once $\mathbf{AC}^0$ formula without changing its depth. Since fooling ϕ is equivalent to fooling $\neg\phi$, by Lemma 20, we may assume that ϕ is a depth-$(d+1)$ read-once NAND formula. Since $s = tr$, we can write $H_d^{\circ s} = (H_d^{\circ t})^{\circ r}$. Consider drawing independent samples $X_1, \ldots, X_r \sim H_d^{\circ t}$. Let $\ell_{\phi,\vec{X}}, u_{\phi,\vec{X}}$ be the formulas guaranteed to us by Lemma 17. For brevity, let $G = G_{\text{MRT}}$, and let $U \sim U_n$ be independent of G and $H_d^{\circ s}$. Let E be the high-probability event of Item 3 of Lemma 17, so whether E occurs depends only on $\vec{X}$. Then,

$$\begin{aligned}
\mathop{\mathbb{E}}_{G_{d+1}}[\phi(G_{d+1})] &= \mathop{\mathbb{E}}_{\vec{X}}\left[\mathop{\mathbb{E}}_{G}[\phi|_{X_1 \circ \cdots \circ X_r}(G)]\right] && \text{By the definition of } G_{d+1} \\
&\leq \mathop{\mathbb{E}}_{\vec{X}}\left[\mathop{\mathbb{E}}_{G}[u_{\phi,\vec{X}}(G)]\right] && \text{By Item 1 of Lemma 17} \\
&\leq \mathop{\mathbb{E}}_{\vec{X}}\left[\mathop{\mathbb{E}}_{G}[u_{\phi,\vec{X}}(G)] \;\middle|\; E\right] + \Pr_{\vec{X}}[\neg E] \\
&\leq \mathop{\mathbb{E}}_{\vec{X}}\left[\mathop{\mathbb{E}}_{U}[u_{\phi,\vec{X}}(U)] + \varepsilon_0 \;\middle|\; E\right] + \Pr_{\vec{X}}[\neg E] && \text{By Corollary 19} \\
&\leq \mathop{\mathbb{E}}_{\vec{X}}\left[\mathop{\mathbb{E}}_{U}[u_{\phi,\vec{X}}(U)] + \varepsilon_0\right] + 2\Pr_{\vec{X}}[\neg E] \\
&\leq \mathop{\mathbb{E}}_{\vec{X},U}[u_{\phi,\vec{X}}(U)] + (1+2r)\varepsilon_0 && \text{By Item 3 of Lemma 17} \\
&\leq \mathop{\mathbb{E}}_{\vec{X},U}[\phi|_{X_1 \circ \cdots \circ X_r}(U)] + (1+2r+3s)\varepsilon_0 && \text{By Item 2 of Lemma 17} \\
&\leq \mathbb{E}[\phi] + (1+2r+4s)\varepsilon_0 && \text{By Corollary 5.}
\end{aligned}$$

A completely analogous argument handles the lower bound. To complete the proof of correctness, we verify that with our choice of parameters, the error is bounded by ε:

$$(1+2r+4s)\varepsilon_0 \leq 5s\varepsilon_0 \leq \frac{1+\log\log(n/\theta)}{2\log\log(n/\varepsilon)} \cdot \varepsilon \leq \varepsilon.$$

Seed Length. Let $q(n,d,\varepsilon)$ denote the seed length of our ε-PRG for depth-d read-once $\mathbf{AC}^0$. We will prove by induction on d that

$$q(n,d,\varepsilon) \leq \log(n/\varepsilon) \cdot (Cd\log\log(n/\varepsilon))^{2d+2}, \tag{6}$$

where C is an absolute constant to be specified later.

In the base case $d = 2$, our PRG is just the PRG by Gopalan et al. [22], which has seed length $C_1 \log(n/\varepsilon)(\log\log(n/\varepsilon))^3$ for some absolute constant C_1. Since $2d+2 > 3$, we can ensure that Equation (6) holds by choosing $C > C_1$.

Now, for the inductive step, fix $d \geq 2$ and consider G_{d+1}. We can divide the seed length of G_{d+1} into three components.

- (The inductive seed length.) To sample from $H_d^{\circ s}$, we must draw $2s$ independent samples from G_d. The number of truly random bits required for this process is bounded by $2s \cdot q(n, d, \alpha)$. There is an absolute constant C_2 so that $s \leq (C_2 d \log\log(n/\varepsilon))^2$. By induction and our choice of $\alpha = \varepsilon^4/n^3$, the number of truly random bits for this component, q_1, is bounded by

$$q_1 \leq 8\log(n/\varepsilon) \cdot (Cd)^{2d+2} \cdot (2 + \log\log(n/\varepsilon))^{2d+2} \cdot s.$$

  To handle the additive 2 term in the middle, we can bound

$$\begin{aligned}(2 + \log\log(n/\varepsilon))^{2d+2} &= (\log\log(n/\varepsilon))^{2d+2} \cdot \left(1 + \frac{2}{\log\log(n/\varepsilon)}\right)^{2d+2} \\ &\leq (\log\log(n/\varepsilon))^{2d+2} \cdot \exp\left(\frac{4d+4}{\log\log(n/\varepsilon)}\right) \\ &\leq e^8,\end{aligned}$$

  since we assumed $d \leq \log\log(n/\varepsilon)$. Therefore,

$$\begin{aligned}q_1 &\leq 8 \cdot e^8 \cdot \log(n/\varepsilon) \cdot (Cd\log\log(n/\varepsilon))^{2d+2} \cdot (C_2 d \log\log(n/\varepsilon))^2 \\ &\leq \frac{1}{3}\log(n/\varepsilon) \cdot (C(d+1)\log\log(n/\varepsilon))^{2(d+1)+2}\end{aligned}$$

  as long as we choose $C > C_2$.
- (The seed length for D and T.) To sample from $H_d^{\circ s}$, we must also draw $2s$ independent samples from D and T. Using standard constructions [33, 2], the number of truly random bits required for this process, q_2, is $2s \cdot O(k + \log(n/\delta) + \log(1/\gamma))$. For some absolute constant C_3, by our choices of k, δ, γ, this is bounded by

$$\begin{aligned}q_2 &\leq C_3 d^2 \log(n/\varepsilon)\log\log(n/\varepsilon)\log\log n \\ &\leq \frac{1}{3}\log(n/\varepsilon) \cdot (C(d+1)\log\log(n/\varepsilon))^{2(d+1)+2},\end{aligned}$$

  provided $C > C_3$.
- (The seed length for the MRT generator.) Because of our choices for the parameters ε_0 and Δ_0, there is an absolute constant C_4 such that in the construction of G_{d+1}, the seed length q_3 of the distribution G_{MRT} from Corollary 19 satisfies

$$q_3 \leq \log(n/\varepsilon) \cdot (C_4(d+1)\log\log(n/\varepsilon))^{2(d+1)+2}.$$

  Choosing $C > C_4$ ensures

$$q_3 \leq \frac{1}{3}\log(n/\varepsilon) \cdot (C(d+1)\log\log(n/\varepsilon))^{2(d+1)+2}.$$

Summing up q_1, q_2, q_3 completes the proof of Equation (6).

Explicitness. Our PRG construction combines explicit PRGs in a straightforward way, so it is explicit as well, i.e., it can be computed in space proportional to its seed length. ◀

9 Fooling Read-Once $\mathrm{AC}^0[\oplus]$ Formulas With a Few Parity Gates

In this section, as outlined in Section 1.3, we prove Theorem 2, which extends our main theorem to the case of $\mathbf{AC}^0[\oplus]$ formulas with a bounded number of parity gates. (An $\mathbf{AC}^0[\oplus]$ *formula* is defined just like an $\mathbf{AC}^0$ formula except that the gates may be labled $\wedge$, $\vee$, or $\oplus$.) The main challenge in proving Theorem 2 is that the sandwiching argument from Section 6 does not easily generalize. The trouble is that the parity function is not monotone, so it does not compose well with sandwiching formulas. This difficulty already arises in the special case of $\mathrm{PARITY} \circ \mathbf{AC}^0$, i.e., the case that the root gate is a parity gate and there are no other parity gates. Instead of true sandwiching formulas, we merely get the following: For every read-once $\mathrm{PARITY} \circ \mathbf{AC}^0$ formula ϕ, there is a $\mathrm{PARITY} \circ \mathbf{AC}^0$ formula $\widetilde{\phi}$ in which every gate rejects with non-negligible probability; this formula $\widetilde{\phi}$ *approximates* ϕ in the sense that

$$\Pr_{X \sim U_n}\left[\phi(X) = \widetilde{\phi}(X)\right] \approx 1.$$

This does not straightforwardly imply correctness of our PRG, because it says nothing about the expectation of ϕ under our *pseudorandom* distribution.

Briefly, to resolve this difficulty, we also design an auxiliary $\mathbf{AC}^0$ formula T_ϕ that certifies that most points x satisfy $\phi(x) = \widetilde{\phi}(x)$. Since T_ϕ is itself fooled by our PRG, $\widetilde{\phi}$ must be a good approximation of ϕ under our pseudorandom distribution as well as the uniform distribution, i.e.,

$$\Pr_{X \sim G_d}\left[\phi(X) = \widetilde{\phi}(X)\right] \approx 1.$$

This condition is a suitable alternative to the sandwiching condition. (A similar approach has been taken in several other works, e.g., [6, 12, 32].)

9.1 Special Case: Read-Once $\mathrm{PARITY} \circ \mathbf{AC}^0$

Toward proving Theorem 2, we begin by considering read-once formulas of the form $\mathrm{PARITY} \circ \mathbf{AC}^0$. Fix any positive integers n, d and any $\varepsilon_1 > 0$. Let $H_d^{\circ s} \circ G_{\mathrm{MRT}}$ be our ε_1-PRG for depth-$(d+1)$ read-once $\mathbf{AC}^0$ formulas used to prove Theorem 1, but with different values for the parameters k, δ, γ (we will explain the changes later). We will prove the following.

▶ **Lemma 21** (Fooling Read-Once $\mathrm{PARITY} \circ \mathbf{AC}^0$)**.** *Let $\phi = \bigoplus_{j=1}^m \phi_j$, where each ϕ_j is a depth-d read-once $\mathbf{AC}^0$ formula, $\phi_1, \ldots, \phi_m$ are on disjoint variable sets, and ϕ is defined over $\{0,1\}^n$. Then $H_d^{\circ 2s} \circ G_{\mathrm{MRT}}$ fools ϕ with error $n^2\varepsilon_1$.*

Note that the PRG in Lemma 21 applies twice as many independent copies of H_d as the PRG in the proof of Theorem 1. Note also that the PRG G_d that underlies H_d is merely assumed to fool depth-d read-once $\mathbf{AC}^0$ formulas (i.e., without any parity gates).

In the remainder of this subsection, we sketch the proof of Lemma 21 by reviewing the proof of Theorem 1 and making the necessary alterations.

9.1.1 H_d Still Preserves the Expectation

The analogue of Corollary 5 still holds in the $\mathrm{PARITY} \circ \mathbf{AC}^0$ setting, with suitable changes to the constants:

▶ **Lemma 22.** *There exist absolute constants $c_1', c_2', c_3' > 0$, such that if we set*

$$k = c_1' \log(nd/\varepsilon_0), \quad \delta = \varepsilon_0 \cdot \left(\frac{c_2'}{\log n}\right)^{-k(2d+2)}, \quad \text{and} \quad \gamma = \frac{c_3'\varepsilon_0}{nd},$$

then H_d satisfies the following. Let $\phi = \bigoplus_{j=1}^{m} \phi_j$, where each ϕ_j is a depth-d read-once NAND *formula, $\phi_1, \ldots, \phi_m$ are on disjoint variable sets, and ϕ is defined on $\{0,1\}^n$. Then, for every integer $t \geq 1$,*

$$\left| \mathop{\mathbb{E}}_{U \sim U_n} [\phi(U)] - \mathop{\mathbb{E}}_{H_d^{\circ t}, V \sim U_n} \left[\phi|_{H_d^{\circ t}}(V) \right] \right| \leq \varepsilon_0 t.$$

Proof sketch. The argument is essentially the same as the proof of Corollary 5. The only change is the width bound. The parity function can be computed by a width-2 ROBP and each ϕ_j can be simulated by a width-$(d+1)$ ROBP, so we can simulate ϕ by a width-$(2d+2)$ ROBP. ◀

9.1.2 $H_d^{\circ t}$ Still Simplifies Formulas Where Each Gate Rejects with Noticeable Probability

Once again, for a formula ϕ as in Lemma 22, we define $\Delta(\phi)$ to be the maximum fan-in of any gate other than the root. The analogue of Lemma 14 also still holds in this setting:

▶ **Lemma 23.** *Let ϕ be as in Lemma 22. Assume $d \leq \log n$, let $\varepsilon_0 > 0$, and let c be the constant guaranteed by Corollary 13. Further assume that $\theta \in (0, \frac{2}{n})$ is such that for every gate ψ in ϕ, possibly excluding the root, $\mathbb{E}[\neg\psi] \geq \theta$. Then, for every integer $t \geq cd \log\log(n/\theta)$ and every $\alpha \leq \frac{\varepsilon_0^2}{8(dn)^2\sqrt{n}\log^2(1/\theta)t}$,*

$$\Pr_{X \sim H_d^{\circ t}} \left[\Delta(\phi|_X) \leq 10\sqrt{\Delta(\phi)}\log^2(1/\theta) \right] \geq 1 - \varepsilon_0,$$

where the PRG for depth-d read-once formulas underlying H_d is instantiated with error α.

Proof. The proof of Lemma 9 still works in this setting, because if ψ is a gate other than the root, then the subformula rooted at ψ is a read-once NAND formula of depth at most d. ◀

9.1.3 Ensuring Noticeable Chance of Rejecting

As discussed at the beginning of this section, we are not able to generalize Lemma 16 to the PARITY$\circ\mathbf{AC}^0$ setting. However, in the original setting of NAND formulas, we can strengthen Lemma 16 by obtaining a read-once $\mathbf{AC}^0$ formula that certifies that the sandwiching formulas are good approximations. Here, for simplicity and because it is sufficient, we focus on the lower sandwiching formula:

▶ **Lemma 24.** *Let ϕ, ε_0, and ℓ_ϕ be as in Lemma 16. There is a depth-d read-once $\mathbf{AC}^0$ formula $T_\phi^\ell \colon \{0,1\}^n \to \{0,1\}$ such that $\mathbb{E}[T_\phi^\ell] \geq 1 - \varepsilon_0$, and for every $x \in \{0,1\}^n$, if $T_\phi^\ell(x) = 1$ then*

$$\ell_\phi(x) = \phi(x).$$

We defer the proof of Lemma 24 to Appendix A, where we prove the generalization involving both the lower and the upper sandwiching formulas (Lemma 30). Just like in Section 6, we must alternately apply Lemma 24 to ensure non-negligible chance of rejection and Lemma 23 to argue that the formula simplifies. The following lemma is analogous to Lemma 17.

▶ **Lemma 25.** *Let ϕ be as in Lemma 22. Assume the parameters $\alpha, k, \delta, \gamma$ underlying H_d satisfy the hypotheses of Lemma 23 and Lemma 22. Let θ be the value in Lemma 16, let t be as in Lemma 23, let $r = \lceil 3 \log \log n \rceil$, and let $s = rt$.*

Sample independent restrictions $X_1, \ldots, X_r \sim H_d^{\circ t}$. For any such vector of restrictions $\vec{X}$, there is a formula $\widetilde{\phi}_{\vec{X}} = \bigoplus_{j=1}^{m} \widetilde{\phi}_j$, where each $\widetilde{\phi}_j$ is a depth-d read-once NAND *formula and $\widetilde{\phi}_1, \ldots, \widetilde{\phi}_m$ are on disjoint variable sets, and there is a function $T_{\phi,\vec{X}} \colon \{0,1\}^n \to \{0,1\}$ with the following properties.*

1. *(Success indication.) For every sample $\vec{X}$ and every point $x \in \{0,1\}^n$, if $T_{\phi,\vec{X}}(x) = 1$, then*

$$\widetilde{\phi}_{\vec{X}}(x) = (\phi|_{X_1 \circ \cdots \circ X_r})(x).$$

2. *(Approximation.) If G ε_1-fools depth-d read-once $\mathbf{AC}^0$ formulas and is independent of $\vec{X}$, then*

$$\mathop{\mathbb{E}}_{\vec{X},G}\left[T_{\phi,\vec{X}}(G)\right] \geq 1 - mr(\varepsilon_1 + (s+1)\varepsilon_0).$$

3. *(Simplicity.) Let $\Delta_0 = 40^4 \log^8(2n/\varepsilon_0)$. Then,*

$$\Pr_{\vec{X}}\left[\Delta\left(\widetilde{\phi}_{\vec{X}}\right) \leq \Delta_0\right] \geq 1 - r\varepsilon_0.$$

The proof of Lemma 25 is similar to the proof of Lemma 17, and we defer it to Appendix B.

9.1.4 G_{MRT} Still Fools Formulas When Δ Is Small

The analogue of Corollary 19 still holds in the PARITY $\circ$ $\mathbf{AC}^0$ setting:

▶ **Lemma 26.** *Fix any positive integers n, d, Δ_0 and any $\varepsilon_0 > 0$. Let ϕ be as in Lemma 22, assume $\Delta(\phi) \leq \Delta_0$, and let G_{MRT} be as in Corollary 19. Then G_{MRT} fools ϕ with error $\varepsilon_0/2$.*

Proof sketch. We can write

$$\phi = \bigoplus_{j=1}^{m} \phi_j = \frac{1}{2} - \frac{1}{2}\prod_{j=1}^{m}(-1)^{\phi_j}.$$

The rest of the argument is the same as in the proof of Corollary 19. ◀

9.1.5 Putting Everything Together for PARITY$\circ$AC^0

Proof Sketch of Lemma 21. We can straightforwardly make each ϕ_j a layered read-once formula without changing its depth. By Lemma 20, either ϕ_j or $\neg\phi_j$ can be computed by a read-once NAND formula with the same underlying tree structure. Furthermore, $\neg$ gates can be pushed upward through $\oplus$ gates. Therefore, since fooling ϕ is the same as fooling $\neg\phi$, we may simply assume that $\phi_1, \ldots, \phi_m$ are NAND formulas.

Since $s = tr$, we can write $H_d^{\circ 2s} = (H_d^{\circ t})^{\circ r} \circ H_d^{\circ s}$. Consider drawing independent samples $X_1, \ldots, X_r \sim H_d^{\circ t}, Y \sim H_d^{\circ s}$. Let $\widetilde{\phi}_{\vec{X}}, T_{\phi,\vec{X}}$ be the functions guaranteed to us by Lemma 25. For brevity, let $G = G_{\text{MRT}}$, and let $U \sim U_n$, all independent of $X_1, \ldots, X_r, Y$. Let E be the high-probability event of Item 3 of Lemma 25, so whether E occurs depends only on $\vec{X}$. Then,

$$\mathop{\mathbb{E}}_{H_d^{\circ 2s},G}[\phi(H_d^{\circ 2s} \circ G)] = \mathop{\mathbb{E}}_{\vec{X},Y,G}\left[\phi|_{X_1 \circ \cdots \circ X_r}(Y \circ G)\right].$$

By Item 2 of Lemma 25,

$$\left| \mathop{\mathbb{E}}_{\vec{X},Y,G} [\phi|_{X_1 \circ \cdots \circ X_r}(Y \circ G)] - \mathop{\mathbb{E}}_{\vec{X},Y,G} \left[\widetilde{\phi}_{\vec{X}}(Y \circ G) \right] \right| \leq \mathop{\mathbb{E}}_{\vec{X},Y,G} \left[\neg T_{\phi,\vec{X}}(Y \circ G) \right].$$

Observe that $Y \circ G$ is exactly the pseudorandom distribution used to prove Theorem 1. Therefore, it ε-fools depth-d read-once $\mathbf{AC}^0$ formulas. Therefore, by Item 1 of Lemma 25,

$$\mathop{\mathbb{E}}_{\vec{X},Y,G} \left[\neg T_{\phi,\vec{X}}(Y \circ G) \right] \leq nr(\varepsilon_1 + (s+1)\varepsilon_0).$$

This will be one term in the overall error. Next, we have

$$\begin{aligned}
&\left| \mathop{\mathbb{E}}_{\vec{X},Y,G} \left[\widetilde{\phi}_{\vec{X}}(Y \circ G) \right] - \mathop{\mathbb{E}}_{\vec{X},Y,U} \left[\widetilde{\phi}_{\vec{X}}(Y \circ U) \right] \right| \\
&\quad \leq \left| \mathop{\mathbb{E}}_{\vec{X}} \left[\mathop{\mathbb{E}}_{Y,G} \left[\left(\widetilde{\phi}_{\vec{X}}|_Y \right)(G) \right] \;\middle|\; E \right] - \mathop{\mathbb{E}}_{\vec{X}} \left[\mathop{\mathbb{E}}_{Y,U} \left[\left(\widetilde{\phi}_{\vec{X}}|_Y \right)(U) \right] \;\middle|\; E \right] \right| + 2 \Pr_{\vec{X}}[\neg E] \\
&\quad \leq \frac{\varepsilon_0}{2} + 2 \Pr_{\vec{X}}[\neg E],
\end{aligned}$$

where the last step was by Lemma 26 (note that $\Delta\left(\widetilde{\phi}_{\vec{X}}|_Y\right) \leq \Delta\left(\widetilde{\phi}_{\vec{X}}\right)$.) This is another term in the overall error. For the next step, by Lemma 22, we have

$$\left| \mathop{\mathbb{E}}_{\vec{X},Y,U} \left[\widetilde{\phi}_{\vec{X}}(Y \circ U) \right] - \mathop{\mathbb{E}}_{\vec{X},U} \left[\widetilde{\phi}_{\vec{X}}(U) \right] \right| \leq s\varepsilon_0.$$

Now, trivially, U fools read-once $\mathbf{AC}^0$ with error 0, so

$$\begin{aligned}
\left| \mathop{\mathbb{E}}_{\vec{X},U} \left[\widetilde{\phi}_{\vec{X}}(U) \right] - \mathop{\mathbb{E}}_{\vec{X},U} [\phi|_{X_1 \circ \cdots \circ X_r}(U)] \right| &\leq \mathop{\mathbb{E}}_{\vec{X},U} \left[\neg T_{\phi,\vec{X}}(U) \right] && \text{By Item 1 of Lemma 25} \\
&\leq nr(s+1)\varepsilon_0 && \text{By Item 2 of Lemma 25.}
\end{aligned}$$

Invoking Lemma 22 one more time gives

$$\left| \mathop{\mathbb{E}}_{\vec{X},U} [\phi|_{X_1 \circ \cdots \circ X_r}(U)] - \mathop{\mathbb{E}}_{U}[\phi(U)] \right| \leq s\varepsilon_0.$$

Adding up all the errors by the triangle inequality, we get

$$\begin{aligned}
\left| \mathop{\mathbb{E}}_{H_d^{\circ 2s},G} [\phi(H_d^{\circ 2s} \circ G)] - \mathop{\mathbb{E}}_{U}[\phi(U)] \right| &\leq nr(\varepsilon + (s+1)\varepsilon_0) + \frac{\varepsilon_0}{2} + 2\Pr[\neg E] \\
&\quad + s\varepsilon_0 + nr(s+1)\varepsilon_0 + s\varepsilon_0 \\
&\leq nr\varepsilon_1 + 5nrs\varepsilon_0 \\
&< n^2\varepsilon_1
\end{aligned}$$

as claimed. ◀

9.2 The General Case of Read-Once $\mathbf{AC}^0[\oplus]$ with t Parity Gates

We first prove a seemingly weak bound on the spectral norm (i.e. the sum of the absolute value of the Fourier coefficients) of a read-once $\mathbf{AC}^0[\oplus]$ formula ϕ in terms of the number of its gates, denoted as $\text{size}(\phi)$.

▶ **Lemma 27.** *Let ϕ be an $\mathbf{AC}^0[\oplus]$ formula. Then,*

$$\left\|\widehat{\phi}\right\|_1 \leq 3^{\mathrm{size}(\phi)}.$$

Proof. The proof uses the fact that spectral norm behaves nicely under composition.

▷ Claim 28. Let $f(x) = g(h_1(x), ..., h_m(x))$, where $f\colon \{0,1\}^n \to \{-1,1\}$, $g\colon \{-1,1\}^m \to \{-1,1\}$. Then,

$$\left\|\widehat{f}\right\|_1 \leq \|\widehat{g}\|_1 \cdot \prod_{i=1}^{m} \left\|\widehat{h_i}\right\|_1$$

Proof. Note that,

$$f(x) = \sum_{S\subseteq[n]} \widehat{g}(S) \prod_{i\in S} h_i(x).$$

The triangle inequality and submultiplicativity of the spectral norm give

$$\left\|\widehat{f}\right\| \leq \sum_{S\subseteq[n]} |\widehat{g}(S)| \prod_{i\in S} \left\|\widehat{h_i}\right\|_1 \leq \sum_{S\subseteq[n]} |\widehat{g}(S)| \prod_{i=1}^{m} \left\|\widehat{h_i}\right\|_1 = \|\widehat{g}\|_1 \cdot \prod_{i=1}^{m} \left\|\widehat{h_i}\right\|_1,$$

where the second inequality uses the fact that $\left\|\widehat{h_i}\right\|_1 \geq 1$, as can be seen as follows. Choose an arbitrary $x \in \{0,1\}^n$, we have

$$1 = |h_i(x)| = \left|\sum_{S\subseteq[n]} \widehat{h_i}(S)\chi_S(x)\right| \leq \sum_{S\subseteq[n]} \left|\widehat{h_i}(S)\right| = \left\|\widehat{h_i}\right\|_1. \quad ◀$$

Let $\wedge_m, \vee_m, \oplus_m\colon \{0,1\}^m \to \{0,1\}$ denote an $\wedge$ gate with m inputs, an $\vee$ gate with m inputs, and a $\oplus$ gate with m inputs respectively. We use the fact that for any $m > 0$,

$$\left\|\widehat{(-1)^{\wedge_m}}\right\|_1, \left\|\widehat{(-1)^{\vee_m}}\right\|_1, \left\|\widehat{(-1)^{\oplus_m}}\right\|_1 \leq 3.$$

Let $\mathcal{G}$ denote the set of the gates in the circuit ϕ. Applying Claim 28 recursively over all the gates of ϕ implies that

$$\left\|\widehat{\phi}\right\|_1 \leq \frac{1}{2} + \frac{1}{2} \cdot \left\|\widehat{(-1)^{\phi}}\right\|_1 \leq \frac{1}{2} + \frac{1}{2} \cdot \prod_{g\in\mathcal{G}} \left\|\widehat{(-1)^{g}}\right\| \leq \frac{1}{2} + \frac{1}{2} \cdot 3^{|\mathcal{G}|} \leq 3^{\mathrm{size}(\phi)}. \quad ◀$$

▶ **Proposition 29.** *Let ϕ be a depth-$(d+1)$ read-once $\mathbf{AC}^0[\oplus]$ formula with $t \geq 1$ parity gates. Then $H_d^{\circ 2s} \circ G_{\mathrm{MRT}}$ fools f with error $n^2\varepsilon_1 \cdot 3^{(d+1)t}$.*

Proof. Let A denote the set of all gates of ϕ that are either a parity gate or have a descendant that is a parity gate. It is easy to see that $|A| \leq (d+1)t$, since each parity gate contributes to at most $d+1$ ancestors. Define $Y = \{y_1, ..., y_m\}$ to be the set of all nodes outside A that have an immediate parent in A, moreover, let $h_1, ..., h_m$ to be the functions computed at these nodes respectively. It is easy to see that

$$\phi(x) = g(h_1, ..., h_m),$$

where g is a depth-d read-once $\mathbf{AC}^0[\oplus]$ formula of size at most $(d+1)t$. Using the Fourier expansion of g,

$$\phi(x) = \sum_{S\subseteq[m]} \widehat{g}(S) \cdot \prod_{i\in S} (-1)^{h_i} = \sum_{S\subseteq[m]} \widehat{g}(S) \cdot (1 - 2 \cdot \bigoplus_{i\in S} h_i).$$

By Lemma 27, $\|\widehat{g}\|_1 \leq 3^{(d+1)t}$, and by Lemma 21, each $\bigoplus_{i\in S} h_i$ is $n^2\varepsilon$ fooled by $H_d^{\circ 2s} \circ G_{\mathrm{MRT}}$. As a result $H_d^{\circ 2s} \circ G_{\mathrm{MRT}}$ fools ϕ with error at most $2 \cdot n^2\varepsilon_1 \cdot 3^{(d+1)t}$. ◀

Proof of Theorem 2. By Proposition 29, the generator $H_d^{\circ 2s} \circ G_{\mathrm{MRT}}$ ε-fools depth-$(d+1)$ read-once $\mathbf{AC}^0[\oplus]$ formulas with at most t parity gates provided we set $\varepsilon_1 := \frac{\varepsilon}{n^2 \cdot 3^{(d+1)t}}$. Now we bound the seed length. The seed length for the distributions D and T underlying H_d is still bounded by $O(d^2 \log(n/\varepsilon_1) \log\log(n/\varepsilon_1) \log\log n)$, just as in the proof of Theorem 1. Similarly, the seed length for G_d and G_{MRT} is still bounded by

$$\log(n/\varepsilon_1) \cdot O((d+1)\log\log(n/\varepsilon_1))^{2(d+1)+2}$$
$$((d+1)t + \log(n/\varepsilon)) \cdot O((d+1)(\log\log(n/\varepsilon) + \log((d+1)t))^{2(d+1)+2}.$$

This second term dominates. Replacing d with $d-1$ completes the proof. ◀

References

1 Miklos Ajtai and Avi Wigderson. Deterministic simulation of probabilistic constant depth circuits. *Advances in Computing Research*, 5(199-222):1, 1989.

2 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.

3 Louay M. J. Bazzi. Polylogarithmic independence can fool DNF formulas. *SIAM J. Comput.*, 38(6):2220–2272, 2009. doi:10.1137/070691954.

4 Andrej Bogdanov, Zeev Dvir, Elad Verbin, and Amir Yehudayoff. Pseudorandomness for width-2 branching programs. *Theory of Computing*, 9:283–292, 2013. doi:10.4086/toc.2013.v009a007.

5 Andrej Bogdanov, Periklis A Papakonstaninou, and Andrew Wan. Pseudorandomness for read-once formulas. In *Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science (FOCS 2011)*, pages 240–246. IEEE, 2011.

6 Mark Braverman. Poly-logarithmic Independence Fools $\mathbf{AC}^0$ Circuits. In *Proceedings of the 24th Annual IEEE Conference on Computational Complexity (CCC 2009)*, pages 3–8. IEEE, 2009.

7 Mark Braverman, Anup Rao, Ran Raz, and Amir Yehudayoff. Pseudorandom Generators for Regular Branching Programs. *SIAM Journal on Computing*, 43(3):973–986, 2014.

8 Suresh Chari, Pankaj Rohatgi, and Aravind Srinivasan. Improved algorithms via approximations of probability distributions. *J. Comput. System Sci.*, 61(1):81–107, 2000. doi:10.1006/jcss.1999.1695.

9 Arkadev Chattopadhyay and Kristoffer Arnsfelt Hansen. Lower bounds for circuits with few modular and symmetric gates. In *Automata, languages and programming*, volume 3580 of *Lecture Notes in Comput. Sci.*, pages 994–1005. Springer, Berlin, 2005. doi:10.1007/11523468_80.

10 Eshan Chattopadhyay, Pooya Hatami, Kaave Hosseini, and Shachar Lovett. Pseudorandom generators from polarizing random walks. In *Proceedings of the 33rd Annual Computational Complexity Conference (CCC 2018)*, volume 102 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 1, 21. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2018.

11 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 Symposium on Theory of Computing (STOC 2018)*, pages 363–375, New York, NY, USA, 2018. ACM. doi:10.1145/3188745.3188800.

12 Eshan Chattopadhyay and David Zuckerman. Explicit two-source extractors and resilient functions. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing*, pages 670–683. ACM, 2016.

13 Sitan Chen, Thomas Steinke, and Salil Vadhan. Pseudorandomness for read-once, constant-depth circuits. *arXiv preprint*, 2015. arXiv:1504.04675.

14 Anindya De. Pseudorandomness for permutation and regular branching programs. In *Proceedings of the 26th Annual IEEE 26th Annual Conference on Computational Complexity (CCC 2011)*, pages 221–231. IEEE, 2011.

15 Anindya De, Omid Etesami, Luca Trevisan, and Madhur Tulsiani. Improved pseudorandom generators for depth 2 circuits. In *Approximation, randomization, and combinatorial optimization*, volume 6302 of *Lecture Notes in Comput. Sci.*, pages 504–517. Springer, Berlin, 2010. doi:10.1007/978-3-642-15369-3_38.

16 Bill Fefferman, Ronen Shaltiel, Christopher Umans, and Emanuele Viola. On beating the hybrid argument. *Theory Comput.*, 9:809–843, 2013. doi:10.4086/toc.2013.v009a026.

17 Michael A. Forbes and Zander Kelley. Pseudorandom Generators for Read-Once Branching Programs, in any Order. In *Proceedings of the 59th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2018)*. IEEE, 2018.

18 Merrick Furst, James B Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical systems theory*, 17(1):13–27, 1984.

19 Dmitry Gavinsky, Shachar Lovett, and Srikanth Srinivasan. Pseudorandom Generators for Read-Once $\mathbf{ACC}^0$. In *Proceedings of the 27th Annual IEEE Conference on Computational Complexity (CCC 2012)*, pages 287–297, 2012.

20 Oded Goldreich and Avi Widgerson. On derandomizing algorithms that err extremely rarely. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC 2014)*, pages 109–118. ACM, New York, 2014.

21 Parikshit Gopalan, Raghu Meka, and Omer Reingold. DNF sparsification and a faster deterministic counting algorithm. *Comput. Complexity*, 22(2):275–310, 2013. doi:10.1007/s00037-013-0068-6.

22 Parikshit Gopalan, Raghu Meka, Omer Reingold, Luca Trevisan, and Salil Vadhan. Better pseudorandom generators from milder pseudorandom restrictions. In *Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science (FOCS 2012)*, pages 120–129. IEEE, 2012.

23 Elad Haramaty, Chin Ho Lee, and Emanuele Viola. Bounded independence plus noise fools products. *SIAM J. Comput.*, 47(2):493–523, 2018. doi:10.1137/17M1129088.

24 Prahladh Harsha and Srikanth Srinivasan. On polynomial approximations to AC^0. *Random Structures & Algorithms*, 54(2):289–303, 2019. doi:10.1002/rsa.20786.

25 Johan Haståd. Almost optimal lower bounds for small depth circuits. In *Proceedings of the eighteenth Annual ACM Symposium on Theory of Computing (STOC 1986)*, pages 6–20. ACM, 1986.

26 Russell Impagliazzo, Raghu Meka, and David Zuckerman. Pseudorandomness from shrinkage. In *Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science (FOCS 2012)*, pages 111–119. IEEE, 2012.

27 Adam R. Klivans, Homin Lee, and Andrew Wan. Mansour's Conjecture is True for Random DNF Formulas. In *Proceedings of the 23rd Annual Conference on Learning Theory (COLT 2010)*, 2010.

28 Michal Koucký, Prajakta Nimbhorkar, and Pavel Pudlák. Pseudorandom generators for group products. In *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing (STOC 2011)*, pages 263–272. ACM, New York, 2011. doi:10.1145/1993636.1993672.

29 Nathan Linial and Noam Nisan. Approximate inclusion-exclusion. *Combinatorica*, 10(4):349–365, 1990.

30 Shachar Lovett and Srikanth Srinivasan. Correlation bounds for poly-size $\mathbf{AC}^0$ circuits with $n^{1-o(1)}$ symmetric gates. In *Approximation, randomization, and combinatorial optimization*, volume 6845 of *Lecture Notes in Comput. Sci.*, pages 640–651. Springer, Heidelberg, 2011. doi:10.1007/978-3-642-22935-0_54.

31 Michael Luby, Boban Velickovic, and Avi Wigderson. Deterministic approximate counting of depth-2 circuits. In *Proceedings of the 2nd Annual Israel Symposium on Theory and Computing Systems (ISTCS 1993)*, pages 18–24. IEEE, 1993.

32 Raghu Meka, Omer Reingold, and Avishay Tal. Pseudorandom Generators for Width-3 Branching Programs. In *Proceedings of the 51st Annual ACM Symposium on Theory of Computing (STOC 2019)*, 2019. To appear.

33 Joseph Naor and Moni Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM Journal on Computing*, 22(4):838–856, 1993.

34 Noam Nisan. Pseudorandom bits for constant depth circuits. *Combinatorica*, 11(1):63–70, 1991. doi:10.1007/BF01375474.

35 Noam Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992.

36 Alexander Razborov. A simple proof of Bazzi's theorem. *ACM Transactions on Computation Theory (TOCT)*, 1(1):3, 2009.

37 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.

38 Rocco A. Servedio and Li-Yang Tan. Improved pseudorandom generators from pseudorandom multi-switching lemmas. *arXiv preprint*, 2018. arXiv:1801.03590.

39 Rocco A. Servedio and Li-Yang Tan. Pseudorandomness for read-k DNF formulas. In *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2019)*, pages 621–638, 2019. doi:10.1137/1.9781611975482.39.

40 Jiří Šíma and Stanislav Žák. Almost k-wise independent sets establish hitting sets for width-3 1-branching programs. In *Computer science—theory and applications*, volume 6651 of *Lecture Notes in Comput. Sci.*, pages 120–133. Springer, Heidelberg, 2011. doi:10.1007/978-3-642-20712-9_10.

41 Thomas Steinke. Pseudorandomness for Permutation Branching Programs Without the Group Theory. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 19, 2012. Report No. 83.

42 Avishay Tal. Tight Bounds on the Fourier Spectrum of $\mathbf{AC}^0$. In Ryan O'Donnell, editor, *Proceedings of the 32nd Annual 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.

43 Luca Trevisan and Tongke Xue. A derandomized switching lemma and an improved derandomization of $\mathbf{AC}^0$. In *Proceedings of the 28th Annual IEEE Conference on Computational Complexity (CCC 2013)*, pages 242–247. IEEE, 2013.

44 Emanuele Viola. Pseudorandom bits for constant-depth circuits with few arbitrary symmetric gates. *SIAM J. Comput.*, 36(5):1387–1403, 2006/07. doi:10.1137/050640941.

A Proofs of Lemma 16 and Lemma 24

Recall that Lemma 16 states that every read-once NAND formula can be sandwiched by two similar structured NAND formulas where every gate has a non-negligible chance of rejecting. We now present the proof of Lemma 16. We emphasize that this argument was already given by Chen, Steinke, and Vadhan [13]; we are reproducing it here to verify the exact parameters of Lemma 16 and so that we can reference the proof when proving Lemma 24.

Proof of Lemma 16. We proceed by induction on $\mathrm{size}(\phi)$, i.e., the number of NAND gates, to prove the lemma with the modified bound $\mathbb{E}[u_\phi - \ell_\phi] \le n\sqrt{\theta} + \mathrm{size}(\phi)\theta$. In the base case $\mathrm{size}(\phi) = 0$, if ϕ is non-constant, it is a single literal, which has expectation $\frac{1}{2}$, so we can simply take $\ell_\phi = u_\phi = \phi$. Now for the inductive step, suppose $\phi = \mathrm{NAND}(\phi_1, \dots, \phi_m)$. Let n_i be the number of inputs to ϕ_i, so $\sum_i n_i = n$ (recall ϕ is read-once). By induction, for each $i \in [m]$, there exist formulas $\ell_{\phi_i} \le \phi_i \le u_{\phi_i}$ with the following properties:

- $\mathbb{E}[u_{\phi_i} - \ell_{\phi_i}] \le n_i\sqrt{\theta} + \mathrm{size}(\phi_i)\theta$.
- Each of u_{ϕ_i} and ℓ_{ϕ_i} has an underlying tree structure that is a subgraph of the underlying tree structure of ϕ_i.
- Every non-constant gate ψ in either ℓ_{ϕ_i} or u_{ϕ_i} satisfies $\mathbb{E}[\psi] \ge \theta$ and $\mathbb{E}[\neg\psi] \ge \theta$.

We consider two cases. For the first case, suppose $\mathbb{E}[\neg\phi] \geq \theta$. In this case, define

$$\ell_\phi = \mathrm{NAND}(u_{\phi_1}, \ldots, u_{\phi_m})$$
$$u_\phi = \begin{cases} \mathrm{NAND}(\ell_{\phi_1}, \ldots, \ell_{\phi_m}) & \text{if that gives } \mathbb{E}[\neg u_\phi] \geq \theta \\ 1 & \text{otherwise.} \end{cases}$$

Because NAND is anti-monotone, $\ell_\phi \leq \phi \leq u_\phi$. In the first case of the definition of u_ϕ, by the union bound, we have

$$\mathbb{E}[u_\phi - \ell_\phi] \leq \sum_{i=1}^{m} (n_i\sqrt{\theta} + \mathrm{size}(\phi_i)\theta) = n\sqrt{\theta} + (\mathrm{size}(\phi) - 1)\theta$$

as desired. In the second case of the definition of u_ϕ, the error only increases by at most θ, which is still within the bound of $n\sqrt{\theta} + \mathrm{size}(\phi)\theta$. Finally, we must verify that every non-constant gate ψ in these formulas satisfies $\mathbb{E}[\psi] \geq \theta$ and $\mathbb{E}[\neg\psi] \geq \theta$. For gates other than the output gate, this is true by induction, so let us verify that it holds for the output gates. We have $\mathbb{E}[\neg\ell_\phi] \geq \mathbb{E}[\neg\phi] \geq \theta$. On the other hand, if ℓ_ϕ is non-constant, then some child u_{ϕ_i} is non-constant, hence $\mathbb{E}[\ell_\phi] \geq \mathbb{E}[\neg u_{\phi_i}] \geq \theta$. Similarly, by construction, if u_ϕ is non-constant, then $\mathbb{E}[\neg u_\phi] \geq \theta$ and $\mathbb{E}[u_\phi] \geq \mathbb{E}[\neg\ell_{\phi_i}] \geq \theta$.

Now, for the second case, suppose $\mathbb{E}[\neg\phi] < \theta$. In this case, define

$$\widetilde{\ell}_\phi = \mathrm{NAND}(u_{\phi_1}, \ldots, u_{\phi_m})$$
$$u_\phi = 1.$$

As before, $\widetilde{\ell}_\phi \leq \phi \leq u_\phi$, and if $\widetilde{\ell}_\phi$ is non-constant, then $\mathbb{E}[\widetilde{\ell}_\phi] \geq \mathbb{E}[\neg u_{\phi_i}] \geq \theta$. Furthermore, $\mathbb{E}[u_\phi - \widetilde{\ell}_\phi] \leq n\sqrt{\theta} + \mathrm{size}(\phi)\theta$. So if $\mathbb{E}[\neg\widetilde{\ell}_\phi] \geq \theta$, we can just set $\ell_\phi = \widetilde{\ell}_\phi$ and we're done. Assume, therefore, that $\mathbb{E}[\neg\widetilde{\ell}_\phi] < \theta$.

In this case, we divide into two subcases. First, suppose that for some i, we have $\mathbb{E}[u_{\phi_i}] \leq \sqrt{\theta}$. Then we define $\ell_\phi = \mathrm{NAND}(u_{\phi_i})$. Clearly, we still have $\ell_\phi \leq \phi$. Furthermore,

$$\mathbb{E}[u_\phi - \ell_\phi] = \mathbb{E}[\neg\ell_\phi] = \mathbb{E}[u_{\phi_i}] \leq \sqrt{\theta}.$$

For the second and final subcase, suppose that for every i, $\mathbb{E}[u_{\phi_i}] > \sqrt{\theta}$. In this case, since $\prod_{i=1}^{m} \mathbb{E}[u_{\phi_i}] = \mathbb{E}[\neg\widetilde{\ell}_\phi] < \theta$, there must be some j such that

$$\theta \leq \prod_{i=1}^{j} \mathbb{E}[u_{\phi_i}] \leq \sqrt{\theta}.$$

Therefore, define

$$\ell_\phi = \mathrm{NAND}(u_{\phi_1}, \ldots, u_{\phi_j}).$$

That way, $\ell_\phi \leq \phi \leq u_\phi$, and $\mathbb{E}[\neg\ell_\phi] \geq \theta$, and

$$\mathbb{E}[u_\phi - \ell_\phi] = \mathbb{E}[\neg\ell_\phi] \leq \sqrt{\theta}.$$

That completes the induction. To get the parameters claimed in the lemma statement, just observe that $\mathrm{size}(\phi) \leq nd$ and $n\sqrt{\theta} + nd\theta < \varepsilon_0$. ◀

Now we state and prove a strengthening of Lemma 24.

▶ **Lemma 30.** *Let ϕ be a depth-d read-once* NAND *formula over n variables with $d \leq n$ and let $\varepsilon_0 > 0$. Let ℓ_ϕ and u_ϕ be the read-once* NAND *formulas guaranteed to us by Lemma 16. Then, there exist $T^\ell_\phi, T^u_\phi \colon \{0,1\}^n \to \{0,1\}$ satisfying the following conditions:*

1. *If $x \in \{0,1\}^n$ is such that $\phi(x) \neq \ell_\phi(x)$ then $T^\ell_\phi(x) = 0$.*
2. *If $x \in \{0,1\}^n$ is such that $\phi(x) \neq u_\phi(x)$ then $T^u_\phi(x) = 0$.*
3. *Both $\mathbb{E}[T^\ell_\phi] \geq 1 - \varepsilon_0$ and $\mathbb{E}[T^\ell_\phi] \geq 1 - \varepsilon_0$.*
4. *Both T^ℓ_ϕ and T^u_ϕ are computable by depth-d read-once $\mathbf{AC}^0$ formulas.*

Roughly speaking, the lemma gives us an "error-indicator" read-once formula that is guaranteed to be zero whenever the sandwiching formula does not give the same value as the original formula. The proof of the lemma will heavily use the proof of Lemma 16.

Proof. The proof is by induction on $\mathrm{size}(\phi)$, as in Lemma 16. In the base case $\mathrm{size}(\phi) = 0$, we simply take $T^\ell_\phi = T^u_\phi = 1$ since $\ell_\phi = u_\phi = \phi$. For the inductive step, suppose $\phi = \mathrm{NAND}(\phi_1, \ldots, \phi_m)$ where for each i, $\mathrm{size}(\phi_i) = n_i$ so that $\sum_i n_i = n$. By our hypothesis, for every $i \in [m]$ there exist formulas ℓ_{ϕ_i} and u_{ϕ_i} guaranteed to us by Lemma 16, as well as formulas $T^u_{\phi_i}$ and $T^u_{\phi_i}$ with the following properties:

- $T^\ell_{\phi_i}(x) = 0$ whenever $\phi_i(x) \neq \ell_{\phi_i}$.
- $T^u_{\phi_i}(x) = 0$ whenever $\phi_i(x) \neq u_{\phi_i}$.
- $\mathbb{E}[\neg T^\ell_{\phi_i}] \leq n_i\sqrt{\theta} + \mathrm{size}(\phi_i)\theta$ and $\mathbb{E}[\neg T^u_{\phi_i}] \leq n_i\sqrt{\theta} + \mathrm{size}(\phi_i)\theta$, for $\theta = \frac{\varepsilon_0^2}{4n^2}$.
- $T^\ell_{\phi_i}$ and $T^u_{\phi_i}$ are computable by depth-$(d-1)$ read-once $\mathbf{AC}^0$ formulas.

Let us first handle T^u_ϕ. For u_ϕ there are two possibilities. It can be either set to $u_\phi = 1$ or set to $u_\phi = \mathrm{NAND}(\ell_{\phi_1}, \ldots, \ell_{\phi_m})$.

1. In the first case, where $u_\phi = 1$, we set $T^u_\phi = \phi$ and so when $T^u_\phi(x) = 1$ clearly $\phi(x) = u_\phi(x) = 1$. To bound $\mathbb{E}[T^u_\phi] = \mathbb{E}[\phi]$, recall that this case is invoked only when either $\mathbb{E}[\neg\phi] < \theta$, in which case the bound is clear, or when $\mathbb{E}[\ell_{\phi_1} \wedge \ldots \wedge \ell_{\phi_m}] = \prod_i \mathbb{E}[\ell_{\phi_i}] < \theta$. In the latter case, since $\mathbb{E}[\phi_i - \ell_{\phi_i}] \leq n_i\sqrt{\theta} + \mathrm{size}(\phi_i)\theta \triangleq \zeta_i$, we obtain

$$\begin{aligned}
\mathbb{E}[\neg T^u_\phi] &= \prod_{i=1}^m \mathbb{E}[\phi_i] \\
&= \prod_{i=1}^m \mathbb{E}[\ell_{\phi_i}] + \sum_{i=1}^m \left((\mathbb{E}[\phi_i] - \mathbb{E}[\ell_{\phi_i}]) \prod_{j=1}^{i-1} \mathbb{E}[\phi_j] \prod_{j=i+1}^{m} \mathbb{E}[\ell_{\phi_j}] \right) \\
&\leq \theta + \sum_{i=1}^m \zeta_i = \theta + n\sqrt{\theta} + (\mathrm{size}(\phi) - 1)\theta = n\sqrt{\theta} + \mathrm{size}(\phi)\theta \leq \varepsilon_0.
\end{aligned}$$

2. In the second case, where $u_\phi = \mathrm{NAND}(\ell_{\phi_1}, \ldots, \ell_{\phi_m})$, set $T^u_\phi = \bigwedge_{i=1}^m T^\ell_{\phi_i}$. If $x \in \{0,1\}^n$ is such that $T^u_\phi(x) = 1$ then $T^\ell_{\phi_i}(x) = 1$ for every $i \in [m]$ and so $u_\phi(x) = \mathrm{NAND}(\phi_1(x), \ldots, \phi_m(x)) = \phi(x)$. To bound $\mathbb{E}[T^u_\phi]$, note that

$$\Pr[T^u_\phi = 0] \leq \sum_{i=1}^m \Pr[T^\ell_{\phi_i} = 0] \leq \sum_{i=1}^m (n_i\sqrt{\theta} + \mathrm{size}(\phi_i)\theta) = n\sqrt{\theta} + (\mathrm{size}(\phi) - 1)\theta \leq \varepsilon_0,$$

   as desired.

In both cases, the depth requirement is immediate.

We shall now handle T^ℓ_ϕ. The two possibilities for ℓ_ϕ are as follows.

1. In the first case, $\ell_\phi = \mathrm{NAND}(u_{\phi_1}, \ldots, u_{\phi_m})$. Here, we set $T^\ell_\phi = \bigwedge_{i=1}^m T^u_{\phi_i}$ and the correctness is similar to case (2) of T^u_ϕ.

2. In the second case, up to reordering of the formulas, there exists $j \in [m-1]$ such that $\ell_\phi = \mathrm{NAND}(u_{\phi_1}, \ldots, u_{\phi_j})$. We choose $T^\ell_\phi = \ell_\phi$, and surely if $x \in \{0,1\}^n$ is such that $\ell_\phi(x) = 1$ then $\phi(x) = 1$ since $\phi \geq \ell_\phi$.
   To bound $\mathbb{E}[T^\ell_\phi]$, recall that j is chosen (again, up to reordering) so that $\mathbb{E}[u_{\phi_1} \wedge \ldots \wedge u_{\phi_j}] \leq \sqrt{\theta}$. Thus, $\mathbb{E}[\neg T^\ell_\phi] \leq \sqrt{\theta} \leq \varepsilon_0$.

Again, in both cases, the depth requirement is immediate. ◀

B Proof of Lemma 25

Toward proving Lemma 25, fix ϕ, define $X_0 = \star^n$, and define $\ell^{(0)}_{j,\vec{X}} = \phi_j$ for each $j \in [m]$. Then, for $i < r$, inductively define

$$\ell^{(i+1)}_{j,\vec{X}} = \ell_{(\ell^{(i)}_{j,\vec{X}}|_{X_i})}$$

That is, $\ell^{(i+1)}_{j,\vec{X}}$ is the lower sandwiching formula when Lemma 16 is applied to $\ell^{(i)}_{j,\vec{X}}|_{X_i}$. Furthermore, define

$$T^{(i+1)}_{j,\vec{X}} = T^\ell_{(\ell^{(i)}_{j,\vec{X}}|_{X_i})}\big|_{X_{i+1}\circ\cdots\circ X_r}.$$

That is, $T^{(i+1)}_{j,\vec{X}}$ is the success certifier of Lemma 24 for the sandwiching formula $\ell^{(i+1)}_{j,\vec{X}}$, restricted according to $X_{i+1} \circ \cdots \circ X_r$. Finally, define

$$\widetilde{\phi}_{\vec{X}} = \bigoplus_{j=1}^{m} \left(\ell^{(r)}_{j,\vec{X}}|_{X_r}\right)$$
$$T_{\phi,\vec{X}} = \bigwedge_{i=1}^{r}\bigwedge_{j=1}^{m} T^{(i)}_{j,\vec{X}}.$$

Proof of Item 1 of Lemma 25. Fix x and assume $T_{\phi,\vec{X}}(x) = 1$. Fix an arbitrary $j \in [m]$. We'll show by backward induction on i that

$$(\ell^{(r)}_{j,\vec{X}}|_{X_r})(x) = (\ell^{(i)}_{j,\vec{X}}|_{X_i\circ X_{i+1}\circ\cdots\circ X_r})(x). \tag{7}$$

In the base case $i = r$, this is trivial. Now for the inductive step, assume Equation (7) is true for $i+1$, and we'll prove it for i. Since $T_{\phi,\vec{X}}(x) = 1$, we must have $T^{(i+1)}_{\phi,\vec{X}}(x) = 1$. That is, $(T^\ell_{(\ell^{(i)}_{j,\vec{X}}|_{X_i})}|_{X_{i+1}\circ\cdots\circ X_r})(x) = 1$. This implies that

$$\ell^{(i+1)}_{j,\vec{X}}(X_{i+1} \circ \cdots \circ X_r \circ x) = \ell^{(i)}_{j,\vec{X}}(X_i \circ X_{i+1} \circ \cdots \circ X_r \circ x).$$

Applying the induction hypothesis completes the proof of Equation (7). Now, plugging in $i = 0$ to Equation (7), we find that

$$(\ell^{(r)}_{j,\vec{X}}|_{X_r})(x) = (\phi_j|_{X_1\circ\cdots\circ X_r})(x). \tag{8}$$

Since Equation (8) holds for all j simultaneously, we can apply the parity operation from $j = 1$ to m to complete the proof. ◀

Proof of Item 2 of Lemma 25. Fix any arbitrary $i \in [r], j \in [m]$. Let $U \sim U_n$ be independent of $\vec{X}$. We have

$$\begin{aligned}
\mathop{\mathbb{E}}_{\vec{X},G}\left[T^{(i)}_{j,\vec{X}}(G)\right] &\geq \mathop{\mathbb{E}}_{\vec{X},U}\left[T^{(i)}_{j,\vec{X}}(U)\right] - \varepsilon_1 && T^{(i)}_{j,\vec{X}} \text{ is a depth-}d \text{ formula} \\
&= \mathop{\mathbb{E}}_{\vec{X},U}\left[T^{\ell}_{(\ell^{(i-1)}_{j,\vec{X}}|_{X_{i-1}})}(X_i \circ \cdots \circ X_r \circ U)\right] - \varepsilon_1 && \text{By the definition of } T^{(i)}_{j,\vec{X}} \\
&\geq \mathop{\mathbb{E}}_{\vec{X},U}\left[T^{\ell}_{(\ell^{(i-1)}_{j,\vec{X}}|_{X_{i-1}})}(U)\right] - \varepsilon_1 - s\varepsilon_0 && \text{By Corollary 5} \\
&\geq 1 - \varepsilon_1 - (s+1)\varepsilon_0 && \text{By Lemma 24.}
\end{aligned}$$

Taking a union bound over i and j completes the proof. ◀

The proof of Item 3 of Lemma 25 is essentially the same as the proof of Item 3 of Lemma 17 and we omit it.

Fourier and Circulant Matrices Are Not Rigid

Zeev Dvir
Department of Computer Science and Department of Mathematics, Princeton University, NJ, USA
zeev.dvir@gmail.com

Allen Liu
Department of Mathematics, MIT, Cambridge, MA, USA
cliu568@mit.edu

Abstract

The concept of matrix rigidity was first introduced by Valiant in [12]. Roughly speaking, a matrix is rigid if its rank cannot be reduced significantly by changing a small number of entries. There has been extensive interest in rigid matrices as Valiant showed in [12] that rigidity can be used to prove arithmetic circuit lower bounds.

In a surprising result, Alman and Williams showed that the (real valued) Hadamard matrix, which was conjectured to be rigid, is actually not very rigid. This line of work was extended by [3] to a family of matrices related to the Hadamard matrix, but over finite fields. In our work, we take another step in this direction and show that for any abelian group G and function $f : G \to \mathbb{C}$, the matrix given by $M_{xy} = f(x - y)$ for $x, y \in G$ is not rigid. In particular, we get that complex valued Fourier matrices, circulant matrices, and Toeplitz matrices are all not rigid and cannot be used to carry out Valiant's approach to proving circuit lower bounds. This complements a recent result of Goldreich and Tal [5] who showed that Toeplitz matrices are nontrivially rigid (but not enough for Valiant's method). Our work differs from previous non-rigidity results in that those works considered matrices whose underlying group of symmetries was of the form $\mathbb{F}_p^n$ with p fixed and n tending to infinity, while in the families of matrices we study, the underlying group of symmetries can be any abelian group and, in particular, the cyclic group $\mathbb{Z}_N$, which has very different structure. Our results also suggest natural new candidates for rigidity in the form of matrices whose symmetry groups are highly non-abelian.

Our proof has four parts. The first extends the results of [1,3] to generalized Hadamard matrices over the complex numbers via a new proof technique. The second part handles the $N \times N$ Fourier matrix when N has a particularly nice factorization that allows us to embed smaller copies of (generalized) Hadamard matrices inside of it. The third part uses results from number theory to bootstrap the non-rigidity for these special values of N and extend to all sufficiently large N. The fourth and final part involves using the non-rigidity of the Fourier matrix to show that the group algebra matrix, given by $M_{xy} = f(x - y)$ for $x, y \in G$, is not rigid for any function f and abelian group G.

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

Keywords and phrases Rigidity, Fourier matrix, Circulant matrix

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.17

Funding *Zeev Dvir*: Research supported by NSF CAREER award DMS-1451191 and NSF grant CCF-1523816.

1 Introduction

1.1 Background

A major goal in complexity theory is to prove lower bounds on the size and depth of arithmetic circuits that compute certain functions. One specific problem that remains open despite decades of effort is to find functions for which we can show super-linear size lower bounds

for circuits of logarithmic depth. In [12], Valiant introduced the notion of matrix rigidity as a possible method of proving such lower bounds for arithmetic circuits. More precisely, over a field $\mathbb{F}$, an $m \times n$ matrix M is said to be (r, s)-rigid if any $m \times n$ matrix of rank at most r differs from M in at least s entries. Valiant showed that for any linear function $f : \mathbb{F}^n \to \mathbb{F}^n$ that can be computed by an arithmetic circuit of size $O(n)$ and depth $O(\log n)$, the corresponding matrix can be reduced to rank $O(\frac{n}{\log \log n})$ by changing $O(n^{1+\epsilon})$ entries for any $\epsilon > 0$. Thus, to prove a circuit lower bound for a function f, it suffices to lower bound the rigidity of the corresponding matrix at rank $O(\frac{n}{\log \log n})$. We call a matrix Valiant-rigid if it is $\left(O(\frac{n}{\log \log n}), O(n^{1+\epsilon})\right)$-rigid for some $\epsilon > 0$, i.e. sufficiently rigid for Valiant's method to yield circuit lower bounds. Over any infinite field, Valiant shows that almost all $n \times n$ matrices are $(r, (n-r)^2)$-rigid for any r, while over a finite field one can get a similar result with a logarithmic loss in the sparsity parameter. Despite extensive work, explicit constructions of rigid matrices have remained elusive.

Over infinite (or very large) fields, there are ways to construct highly rigid matrices using either algebraically independent entries or entries that have exponentially large description (see [7–9]) [1]. However, these constructions are not considered to be fully explicit as they do not tell us anything about the computational complexity of the corresponding function. Ideally, we would be able to construct rigid $0, 1$-matrices, but even a construction where the entries are in a reasonably simple field (such as the Fourier matrix) would be a major breakthrough. The best known constructions of such matrices are $(r, O(\frac{n^2}{r} \log \frac{n}{r}))$-rigid (see [4, 11]). There has also been work towards constructing semi-explicit rigid matrices, which require $O(n)$ bits of randomness (instead of the usual $O(n^2)$), as such a construction would still yield circuit lower bounds through Valiant's approach [2]. The best result in this realm (see [5]) shows that random Toeplitz matrices are $(r, \frac{n^3}{r^2 \log n})$-rigid with high probability. Note that both of these bounds become trivial when r is $\frac{n}{\log \log n}$.

Many well-known families of matrices, such as Hadamard matrices and Fourier transform matrices, have been conjectured to be Valiant-rigid [10]. However, a recent line of works (see [1, 3]) shows that certain well-structured matrices are not rigid. Alman and Williams show in [1] that the $2^n \times 2^n$ Hadamard matrix, given by $H_{xy} = (-1)^{x \cdot y}$ as x and y range over $\{0, 1\}^n$, is not Valiant-rigid over $\mathbb{Q}$. Along similar lines, Dvir and Edelman show in [3] that group algebra matrices for the additive group $\mathbb{F}_p^n$, given by $M_{xy} = f(x - y)$ where $f : \mathbb{F}_p^n \to \mathbb{F}_p$ and x, y range over $\mathbb{F}_p^n$, are not Valiant-rigid over $\mathbb{F}_p$ (where we view p as fixed and n goes to infinity). The Hadamard matrix and the group algebra matrices for $\mathbb{F}_p^n$ satisfy the property that for any $\epsilon > 0$, there exists an $\epsilon' > 0$ such that it is possible to change at most $N^{1+\epsilon}$ entries and reduce the rank to $N^{1-\epsilon'}$ (where N denotes the size of the matrix). The proofs of both results rely on constructing a matrix determined by a polynomial $P(x, y)$ that agrees with the given matrix on almost all entries and then arguing that the constructed matrix has low rank.

[1] It remains open to construct a matrix that is Valiant-rigid, even if we only require that the entries live in a number field of dimension polynomial in the size of the matrix.

[2] Note however, that it is easy to construct rigid matrices with $O(n^{1+\epsilon})$ bits of randomness for any $\epsilon > 0$ (for example by taking a random matrix with at most n^ϵ non-zeros per row) but this is not sufficient for Valiant's approach.

1.2 Our Contribution

In this paper, we show that several broad families of matrices, including Fourier, circulant and Toeplitz matrices[3], are all not Valiant-rigid. The families of matrices we consider in our work have very different underlying group structure than those considered in previous works. Both [1, 3] analyze matrices constructed from an underlying group of the form $\mathbb{F}_p^n$ with p fixed and n tending to infinity. Fourier and circulant matrices, which we focus on, are analogs of the Hadamard and group algebra matrices[4] for a cyclic group $\mathbb{Z}_N$. Since any abelian group can be decomposed into simple building blocks of the form $\mathbb{Z}_N$, our results extend to all abelian groups (see details below). While most natural constructions of matrices are highly symmetric, our results suggest that matrices that are symmetric under abelian groups are not rigid and that perhaps we should look toward less structured matrices, or matrices whose symmetry group is non-abelian, as candidates for rigidity.

We now move into a more technical overview of our paper. Define the regular-rigidity of a matrix A, $\mathsf{r}_A(r)$, as the minimum value of s such that it is possible to change at most s entries in each row and column of A to obtain a matrix of rank at most r. The notion of regular-rigidity is weaker than the usual notion of rigidity (and is also weaker than the commonly used notion of row-rigidity) as if A is an $n \times n$ matrix and A is (r, ns)-rigid then $\mathsf{r}_A(r) \geq s$. Note that this actually makes our results stronger as we will show that the matrices we consider are not regular-rigid.

All matrices that we deal with will be over $\mathbb{C}$. The $d^n \times d^n$ generalized Hadamard matrix $H_{d,n}$ has rows and columns indexed by $\mathbb{Z}_d^n$ and entries $H_{xy} = \omega^{x \cdot y}$ where $\omega = e^{\frac{2\pi i}{d}}$. Throughout this paper, we use the term Hadamard matrix to refer to any generalized Hadamard matrix. We use $F_N = H_{N,1}$ to denote the $N \times N$ Fourier transform matrix. Our main result, that all Fourier matrices are not rigid enough to carry out Valiant's approach, is stated below.

▶ **Theorem 1** (Fourier Matrices are Not Rigid). *Let F_N denote the $N \times N$ Fourier transform matrix. For any fixed $0 < \epsilon < 0.1$ and N sufficiently large,*

$$r_{F_N}\left(\frac{N}{2^{\epsilon^4 (\log N)^{0.0004}}}\right) \leq N^{9\epsilon}$$

One key idea in our work is the observation that, if a large family of matrices $\mathcal{A}$ are all diagonalizable by a single matrix M then, the rigidity of *any* matrix $A \in \mathcal{A}$ implies the rigidity of the single matrix M. This situation happens, e.g., when $\mathcal{A}$ is the family of circulant matrices and M is the Fourier matrix. This simple, yet crucial observation allows us to deduce the non-rigidity of a larger family of matrices.

▶ **Corollary 2** (Circulant Matrices are not Rigid). *Let $0 < \epsilon < 0.1$ be fixed. For all sufficiently large N, if M is an $N \times N$ circulant matrix over $\mathbb{C}$,*

$$r_M\left(\frac{N}{2^{\epsilon^4 (\log N)^{0.0003}}}\right) \leq N^{20\epsilon}$$

[3] It is not hard to see that rigidity of circulant and Toeplitz matrices is essentially the same question so for the sake of consistency with our (group theoretic) approach we will primarily consider circulant matrices.

[4] While group algebra matrices are supposed to be defined as $M_{xy} = f(x - y)$, we will work with $M_{xy} = f(x + y)$ in the body of our paper for technical reasons. Note that the two definitions differ only in a permutation of the rows and thus have the same rigidity.

Also notice that since any Toeplitz matrix of size at most $\frac{N}{2}$ can be embedded in an $N \times N$ circulant matrix, the above implies an analogous result for all Toeplitz matrices. While [5] shows nontrivial rigidity lower bounds for rank much smaller than N, our results imply that there are actually no nontrivial rigidity lower bounds for rank close to N.

With a bit more work, it is possible to prove the non-rigidity of group algebra matrices for any abelian group.

▶ **Theorem 3.** *Let $0 < \epsilon < 0.1$ be fixed. Let G be an abelian group and $f : G \to \mathbb{C}$ be a function. Let M be a matrix with rows and columns indexed by elements $x, y \in G$ and entries $M_{xy} = f(x - y)$. If $|G|$ is sufficiently large then*

$$r_M\left(\frac{2|G|}{2^{\epsilon^6(\log|G|)^{0.0001}}}\right) \leq |G|^{26\epsilon}$$

1.3 Proof Overview

We now take a more detailed look at the techniques used in the proof of Theorem 1.

1.3.1 Generalized Hadamard Matrices

The first step in the proof of Theorem 1 is proving the following result that all Hadamard matrices are not rigid.

▶ **Theorem 4** (Hadamard Matrices are not Rigid)**.** *For fixed d and $0 < \epsilon < 0.1$, there exists an ϵ' such that for all sufficiently large n, $r_{H_{d,n}}\left(d^{n(1-\epsilon')}\right) \leq d^{n\epsilon}$*

Note that Theorem 4 generalizes the main result of [1] (which only deals with $d = 2$). Also, given any $d^n \times d^n$ matrix of the form $M_{xy} = f(x - y)$ with $f : \mathbb{Z}_d^n \to \mathbb{C}$, we can permute its rows so that it is diagonalized by $H_{d,n}$. Thus, we can apply the diagonalization trick mentioned above and obtain the following result, which extends the work in [3] to matrices over $\mathbb{C}$.

▶ **Corollary 5.** *Let f be a function from $\mathbb{Z}_d^n \to \mathbb{C}$ and let M be a $d^n \times d^n$ matrix with $M_{xy} = f(x - y)$. Then for any fixed d and $0 < \epsilon < 0.1$, there exists an $\epsilon' > 0$ such that for all sufficiently large n, $r_M\left(d^{n(1-\epsilon')}\right) \leq d^{n\epsilon}$*

1.3.2 Fourier Matrices

Equipped with the machinery for Hadamard matrices, we can complete the proof of Theorem 1. Our proof consists of two steps. First we show that for integers N of a very special form, the $N \times N$ Fourier matrix is not rigid because it can be decomposed into submatrices with Hadamard-type structure. We say an integer N is well-factorable if it is a product of distinct primes $q_1, \ldots, q_l$ such that for all i, $q_i - 1$ has no large prime power divisors. We will make this notion more precise later, but informally, the first step is as follows:

▶ **Theorem 6.** *Let F_N denote the $N \times N$ Fourier transform matrix. For any fixed $0 < \epsilon < 0.1$ and well-factorable integer N, we have*

$$r_{F_N}\left(\frac{N}{2^{\epsilon^4(\log N)^{0.0005}}}\right) \leq N^{4\epsilon}$$

The main intuition is that if N is a product of distinct primes $q_1, \ldots, q_l$, then within the Fourier matrix F_N, we can find submatrices whose rows and columns can be indexed by $\mathbb{Z}_{q_1}^* \otimes \cdots \otimes \mathbb{Z}_{q_l}^*$. This multiplicative structure can be replaced by the additive structure of $\mathbb{Z}_{q_1-1} \otimes \cdots \otimes \mathbb{Z}_{q_l-1}$. We can then factor each additive group $\mathbb{Z}_{q_i-1}$ into prime power components. If $q_1 - 1, \ldots, q_l - 1$ all have no large prime power divisors, we expect prime powers to be repeated many times when all of the terms are factored. This allows us to find submatrices with $\mathbb{Z}_d^l$ additive structure for which we can apply tools such as Theorem 4 and Corollary 5 to reduce the rank while changing a small number of entries. We then bound the rank and total number of entries changed over all submatrices to deduce that F_N is not rigid.

The second step of our proof that Fourier matrices are not rigid involves extending Theorem 6 to all values of N. The diagonalization trick gives that $N \times N$ circulant matrices are not rigid when N is well-factorable. We then show that for $N' < \frac{N}{2}$, we can rescale the columns of the $N' \times N'$ Fourier matrix and embed it into an $N \times N$ circulant matrix. As long as N' is not too much smaller than N (say $N' > \frac{N}{(\log N)^2}$), we get that the $N' \times N'$ Fourier matrix is not rigid. Thus, for each well-factorable N and all N' in the range $\frac{N}{(\log N)^2} < N' < \frac{N}{2}$, the $N' \times N'$ Fourier transform matrix is not rigid. We then use a number theoretic result of [2] to show that the gaps between well-factorable integers are not too large. Thus, the above intervals cover all integers as N runs over all well-factorable numbers, finishing the proof.

1.4 Organization

In Section 2, we introduce notation and prove several basic results that we will use throughout the paper. In Section 3, we show that Hadamard and several closely related families of matrices are not rigid. In Section 4, we show that $N \times N$ Fourier matrices are not rigid when N satisfies certain number-theoretic properties. In Section 5, we complete the proof that all Fourier matrices are not rigid. We then deduce that all Toeplitz matrices are not rigid. In Section 6, we use the results from the previous section to show that group algebra matrices for abelian groups are not rigid. Finally, in Section 7, we discuss a few open questions and possible directions for future work.

2 Preliminaries

Throughout this paper, we let $d \geq 2$ be an integer and $\omega = e^{\frac{2\pi i}{d}}$ be a primitive d^{th} root of unity. When we consider an element of $\mathbb{Z}_d^n$, we will view it as an n-tuple with entries in the range $[0, d-1]$. When we say a list of d^n elements $x_1, \ldots, x_{d^n}$ is indexed by $\mathbb{Z}_d^n$, we mean that each x_i is labeled with an element of $\mathbb{Z}_d^n$ such that all labels are distinct and the labels of $x_1, \ldots, x_{d^n}$ are in lexicographical order.

2.1 Basic Notation

We will frequently work with tuples, say $I = (i_1, \ldots, i_n) \in \mathbb{Z}_d^n$. Below we introduce some notation for dealing with tuples that will be used later on.

▶ **Definition 7.** *For a tuple I, we let I^i denote its i^{th} entry. For instance if $I = (i_1, \ldots, i_n)$ then $I^k = i_k$.*

▶ **Definition 8.** *For an n-tuple $I = (i_1, i_2, \ldots, i_n)$, define the polynomial over n variables $x^I = x_1^{i_1} \ldots x_n^{i_n}$.*

▶ **Definition 9.** *For ω a d^{th} root of unity and an n-tuple $I = (i_1, i_2, \ldots, i_n) \in \mathbb{Z}_d^n$, we define $\omega^{[I]} = (\omega^{i_1}, \ldots, \omega^{i_n})$.*

▶ **Definition 10.** *For a function $f : \mathbb{Z}_d^n \to \mathbb{C}$, define the n-variable polynomial P_f as*

$$P_f = \sum_{I \in \mathbb{Z}_d^n} f(I) x^I$$

▶ **Definition 11.** *For an n-tuple $I = (i_1, i_2, \ldots, i_n)$, we define the set $\mathsf{perm}(I)$ to be a set of n-tuples consisting of all distinct permutations of the entries of I. Similarly, for a set of n-tuples S, we define $\mathsf{perm}(S)$ to be the set of all n-tuples that can be obtained by permuting the entries of some element of S.*

▶ **Definition 12.** *We say a set $S \subseteq \mathbb{Z}_d^n$ is symmetric if for any $I \in S$, $\mathsf{perm}(I) \subseteq S$.*

▶ **Definition 13.** *For a set of n-tuples S, let $\mathsf{red}(S)$ denote the set of equivalence classes under permutation of entries in S. Let $\mathsf{rep}(S)$ be a set of n-tuples formed by taking one representative from each equivalence class in $\mathsf{red}(S)$ (note $\mathsf{rep}(S)$ is not uniquely determined but this will not matter for our use).*

Note that if $\mathsf{rep}(S) = \{I_1, \ldots, I_k\}$, then the sets $\mathsf{perm}(I_1), \mathsf{perm}(I_2), \ldots, \mathsf{perm}(I_k)$ are disjoint and their union contains S. If the set S is symmetric then their union is exactly S.

2.2 Special Families of Matrices

We now define notation for working with a few special families of matrices.

▶ **Definition 14.** *An $N \times N$ matrix M is called a Toeplitz matrix if M_{ij} depends only on $i - j$. An $N \times N$ matrix M is called a Hankel matrix if M_{ij} depends only on $i + j$. Note that the rows of any Toeplitz matrix can be permuted to obtain a Hankel matrix so any non-rigidity results we show for one family also hold for the other.*

▶ **Definition 15.** *For an abelian group G and a function $f : G \to \mathbb{C}$, let $M_G(f)$ denote the $|G| \times |G|$ matrix (over $\mathbb{C}$) whose rows and columns are indexed by elements $x, y \in G$ and whose entries are given by $M_{xy} = f(x + y)$. When it is clear what G is from context, we will simply write $M(f)$. We let V_G denote the family of matrices $M_G(f)$ as f ranges over all functions from G to $\mathbb{C}$. We call V_G the family of adjusted group algebra matrices for the group G. When G is a cyclic group, we call the matrices in V_G adjusted-circulant.*

Compared to the usual group algebra (and circulant) matrices defined by $M_{xy} = f(x - y)$, the matrix $M_G(f)$ differs only in a permutation of the rows. In the proceeding sections, we will work with $M_G(f)$ for technical reasons, but it is clear that the same non-rigidity results hold for the usual group algebra matrices. Similarly, we will use adjusted-circulant and Hankel matrices as it is clear that the same non-rigidity results hold for circulant and Toeplitz matrices. Also note that adjusted-circulant matrices are a special case of Hankel matrices.

▶ **Definition 16.** *Let $H_{d,n}$ denote the $d^n \times d^n$ Hadamard matrix, i.e. the matrix whose rows and columns are indexed by n-tuples $I, J \in \mathbb{Z}_d^n$ and whose entries are $H_{IJ} = \omega^{I \cdot J}$ where $\omega = e^{\frac{2\pi i}{d}}$. When $n = 1$, we define $F_d = H_{d,1}$ and call F_d a Fourier matrix.*

2.3 Matrix Rigidity

Here, we review basic notation for matrix rigidity.

▶ **Definition 17.** *For a matrix M and a real number r, we define $R_M(r)$ to be the smallest number s for which there exists a matrix A with at most s nonzero entries and a matrix B of rank at most r such that $M = A + B$. If $R_M(r) \geq s$, we say M is (r, s)-rigid.*

▶ **Definition 18.** *For a matrix M and a real number r, we define $r_M(r)$ to be the smallest number s for which there exists a matrix A with at most s nonzero entries in each row and column and a matrix B of rank at most r such that $M = A + B$. If $r_M(r) \geq s$, we say M is (r, s)-regular rigid.*

It is clear that if a matrix is (r, ns)-rigid, then it must be (r, s)-regular rigid. In proceeding sections, we will show that various matrices are not $(\frac{N}{\log\log N}, N^\epsilon)$-regular rigid for any $\epsilon > 0$ and this will imply that Valiant's method for showing circuit lower bounds in [12] cannot be applied.

2.4 Preliminary Results

Next, we mention several basic results that will be useful in the proofs later on.

▷ Claim 19. $H_{d,n} = \underbrace{F_d \otimes \cdots \otimes F_d}_{n}$ where $\otimes$ denotes the Kronecker product.

Proof. This can easily be verified from the definition. ◁

▷ Claim 20. $H_{d,n} H_{d,n}^* = d^n I$ where $H_{d,n}^*$ is the conjugate transpose of $H_{d,n}$ and I is the identity matrix.

Proof. We verify that $F_d F_d^* = dI$, and then using the previous claim, we deduce that $H_{d,n} H_{d,n}^* = d^n I$. ◁

▷ Claim 21. Let $f : \mathbb{Z}_d^n \to \mathbb{C}$ be a function. Let ω be a d^{th} root of unity and set $P_f = \sum_{I \in \mathbb{Z}_d^n} f(I) x^I$. Let $D = H_{d,n} M_{\mathbb{Z}_d^n}(f) H_{d,n}$. Then D is a diagonal matrix with diagonal entries $d^n P_f(\omega^{[J]})$ as J ranges over $\mathbb{Z}_d^n$.

Proof. First, we analyze the product $M_{\mathbb{Z}_d^n}(f) H_{d,n}$. This is a $d^n \times d^n$ matrix and its rows and columns can naturally be indexed by tuples $I, J \in \mathbb{Z}_d^n$. The entry with row indexed by I and column indexed by J is

$$\sum_{I' \in \mathbb{Z}_d^n} f(I + I') \omega^{I' \cdot J} = \omega^{-I \cdot J} \sum_{I' \in \mathbb{Z}_d^n} f(I + I') \omega^{(I' + I) \cdot J} = \omega^{-I \cdot J} P_f(\omega^{[J]})$$

Therefore, the columns of $M_{\mathbb{Z}_d^n}(f) H_{d,n}$ are multiples of the columns of $H_{d,n}^*$. In fact, the column of $M_{\mathbb{Z}_d^n}(f) H_{d,n}$ indexed by J is $P_f(\omega^{[J]})$ times the corresponding column of $H_{d,n}^*$. Since $H_{d,n} H_{d,n}^* = d^n I$, D must be a diagonal matrix whose entries on the diagonal are $d^n P_f(\omega^{[J]})$ as J ranges over $\mathbb{Z}_d^n$. ◁

Plugging $n = 1$ into the above gives:

▷ Claim 22. Let M be a $d \times d$ adjusted-circulant matrix. Then $F_d M F_d$ is a diagonal matrix.

Claim 21 gives us a characterization of the rank of matrices of the form $M_{\mathbb{Z}_d^n}(f)$.

▷ Claim 23. Let $f : \mathbb{Z}_d^n \to \mathbb{C}$ be a function. Let ω be a d^{th} root of unity and say $P_f = \sum_{I \in \mathbb{Z}_d^n} f(I)x^I$ has C roots among the set $\{(\omega^{i_1}, \ldots, \omega^{i_n}) | (i_1, \ldots, i_n) \in \mathbb{Z}_d^n\}$. Then $\mathsf{rank}(M_{\mathbb{Z}_d^n}(f)) = d^n - C$.

Proof. Consider the product $D = H_{d,n} M_{\mathbb{Z}_d^n}(f) H_{d,n}$. Note that $H_{d,n}$ is clearly invertible by Claim 20. Therefore, it suffices to compute the rank of D. By Claim 21, D must be a diagonal matrix whose entries on the diagonal are $d^n P_f(\omega^{[J]})$ as J ranges over $\mathbb{Z}_d^n$. The rank of D is the number of nonzero diagonal entries which is simply $d^n - C$ ◁

As mentioned in the introduction, we can relate the rigidity of a matrix to the rigidity of matrices that it diagonalizes.

▶ **Lemma 24.** *If $B = A^*DA$ where D is a diagonal matrix and $\mathsf{r}_A(r) \leq s$ then $\mathsf{r}_B(2r) \leq s^2$. The same inequality holds also for $B' = ADA$.*

Proof. Let E be the matrix with at most s nonzero entries in each row and column such that $A - E$ has rank at most r. We have

$$B - E^*DE = A^*D(A - E) + (A^* - E^*)DE$$

Since $\mathsf{rank}(A - E) \leq r$, $\mathsf{rank}(B - E^*DE) \leq 2r$. Also, E^*DE has at most s^2 nonzero entries in each row and column so $\mathsf{r}_B(2r) \leq s^2$. The second part can be proved in the exact same way with A^* replaced by A. ◀

In light of Lemma 24, Claim 22, and Claim 21, proving non-rigidity for $d \times d$ circulant matrices reduces to proving non-rigidty for F_d and proving non-rigidity for group algebra matrices for $\mathbb{Z}_d^n$ reduces to proving non-rigidity for $H_{d,n}$. Below, we show that these statements are actually equivalent.

▷ Claim 25. It is possible to rescale the rows and columns of $H_{d,n}$ to get a matrix of the form $M_{\mathbb{Z}_d^n}(f)$ for some symmetric function $f : \mathbb{Z}_d^n \to \mathbb{C}$. In particular, it is possible to rescale the rows and columns of F_d to get an adjusted-circulant matrix.

Proof. Let ζ be such that $\zeta^2 = \omega$. Multiply each row of $H_{d,n}$ by $\zeta^{(I \cdot I)}$ and each column by $\zeta^{(J \cdot J)}$ to get a matrix H'. We have

$$H'_{IJ} = \zeta^{(I+J)\cdot(I+J)}$$

For a tuple $x = (x_1, \ldots, x_n) \in \mathbb{Z}_d^n$, we define $f(x) = \zeta^{x_1^2 + \cdots + x_n^2}$. To complete the proof, it suffices to show that $f : \mathbb{Z}_d^n \to \mathbb{C}$ is well defined. To do this, we will show that ζ^{x^2} depends only on the residue of $x \mod d$. If d is odd, we can choose ζ to be a d^{th} root of unity and the claim is clear. If d is even $\zeta^{(x+d)^2} = \zeta^{x^2}\zeta^{2dx+d^2}$ but since $2dx + d^2$ is a multiple of $2d$, $\zeta^{2dx+d^2} = 1$ and thus $\zeta^{(x+d)^2} = \zeta^{x^2}$. ◁

3 Non-rigidity of Generalized Hadamard Matrices

In this section, we show that the Hadamard matrix $H_{d,n}$ becomes highly non-rigid for large values of n. The precise result is stated below.

▶ **Theorem 26.** *Let $N = d^n$ for positive integers d, n. Let $0 < \epsilon < 0.1$ and assume $n \geq \frac{d^2(\log d)^2}{\epsilon^4}$. Then $\mathsf{r}_{H_{d,n}}(N^{1 - \frac{\epsilon^4}{d^2 \log d}}) \leq N^{\epsilon}$.*

First we prove a few lemmas about symmetric polynomials that we will use in the proof of Theorem 26.

▶ **Lemma 27.** *Let T_m denote the set of tuples in $\mathbb{Z}_d^n$ such that at least m entries are equal to 0. Say $\mathsf{rep}(T_m) = \{I_1, \ldots, I_k\}$. Consider the polynomials $P_1(x_1, \ldots, x_n), \ldots, P_k(x_1, \ldots, x_n)$ defined by*

$$P_i(x_1, \ldots, x_n) = \sum_{I \in \mathsf{perm}(I_i)} x^I$$

For any complex numbers $y_1, \ldots, y_m$, and any polynomial $Q(x_{m+1}, \ldots x_n)$ that is symmetric and degree at most $d-1$ in each of its variables, there exist coefficients $c_1, \ldots, c_k$ such that

$$Q(x_{m+1}, \ldots, x_n) = \sum c_i P_i(y_1, \ldots y_m, x_{m+1}, \ldots, x_n)$$

Proof. It suffices to prove the statement for all Q of the form

$$\sum_{I'' \in \mathsf{perm}(I')} x^{I''}$$

where $I' \in \mathbb{Z}_d^{n-m}$. We will prove this by induction on the degree. Clearly one of the I_i is $(0, 0 \ldots 0)$, so one of the polynomials $P_i(x_1, \ldots, x_n)$ is constant. This finishes the case when Q has degree 0. Now we do the induction step. Note that we can extend I' to an element of T_m by setting the first m entries equal to 0. Call this extension I and say that $I \in \mathsf{perm}(I_i)$. We have

$$\sum_{I'' \in \mathsf{perm}(I')} x^{I''} = P_i(y_1, \ldots, y_m, x_{m+1}, \ldots, x_n) - R(y_1, \ldots, y_m, x_{m+1}, \ldots x_n)$$

$R(y_1, \ldots, y_m, x_{m+1}, \ldots x_n)$, when viewed as a polynomial in $x_{m+1}, \ldots, x_n$ (since $y_1, \ldots, y_m$ are complex numbers that we can plug in), is symmetric and of lower degree than the left hand side. Thus, using the induction hypothesis, we can write R in the desired form. This completes the induction step. ◀

The key ingredient in the proof of Theorem 26 is the following lemma which closely resembles the main result in [3], but deals with matrices over $\mathbb{C}$ instead of matrices over a finite field.

▶ **Lemma 28.** *Let $f : \mathbb{Z}_d^n \to \mathbb{C}$ be a symmetric function on the n variables. Let $N = d^n$. Let $0 < \epsilon < 0.1$ and assume $n \geq \frac{d^2(\log d)^2}{\epsilon^4}$. Then $r_{M(f)}(N^{1 - \frac{\epsilon^4}{d^2 \log d}}) \leq N^\epsilon$.*

Let $\delta = \epsilon^2$, $m = \lceil n\left(\frac{1-\delta}{d}\right) \rceil$ and let S denote the set of all tuples $(i_1, i_2, \ldots, i_n) \in \mathbb{Z}_d^n$ such that the entries indexed $1, 2, \ldots m$ are equal to 0, the entries indexed $m+1, \ldots, 2m$ are equal to 1 and in general for $0 \leq i \leq d-1$, the entries indexed $im+1, \ldots, (i+1)m$ are equal to i. Note $|S| = d^{n-dm} \approx d^{\delta n} = N^{\epsilon^2}$ (since $n - dm$ is approximately δn).

The main idea will be to change f in a small number of locations so that it has many zeros in the set $\{\omega^{[I]} | I \in \mathbb{Z}_d^n\}$ in order to make use of Claim 23. More precisely, first we will change f to f' by changing its values in at most N^ϵ places so that f' is still symmetric in all of the variables and

$$P_{f'}\left(\omega^{[I]}\right) = 0 \ \forall I \in S$$

Note that although the size of S is small, the fact that f' is symmetric implies that f' also vanishes on $\mathsf{perm}(S)$, which covers almost all of $\mathbb{Z}_d^n$. Once we have shown the above, we quantitatively bound the number of entries changed between $M(f)$ and $M(f')$ and also the rank of $M(f')$ to complete the proof of Lemma 28. To do the first part, we need the following sub-lemma.

▶ **Lemma 29.** *Let T denote the set of all tuples $(i_1, i_2, \ldots, i_n) \in \mathbb{Z}_d^n$ such that at least $n(1-\delta)$ of the entries are 0. By changing the values of f only on elements of T, we can obtain f' satisfying*

$$P_{f'}\left(\omega^{[I]}\right) = 0 \ \forall I \in S \tag{1}$$

Proof. We interpret (1) as a system of linear equations where the unknowns are the values of f' at various points. Let $\mathsf{rep}(T) = \{J_1, J_2, \ldots, J_k\}$ for $J_1, J_2, \ldots J_k \in T$. Since we must maintain that f' is symmetric, there are essentially k variables each corresponding to an equivalence class of tuples under permutations. Each equivalence class is of the form $\mathsf{perm}(J_j)$ and we denote the corresponding variable by m_j. The system of equations in (1) can be rewritten in the form

$$\sum_{j=1}^{k} m_j \sum_{J \in \mathsf{perm}(J_j)} \omega^{I \cdot J} + \sum_{J' \notin T} f(J')\omega^{I \cdot J'} = 0 \ \forall I \in S$$

If we let $\mathsf{rep}(S) = \{I_1, I_2, \ldots, I_l\}$, the system has exactly l distinct equations corresponding to each element of $\mathsf{rep}(S)$ due to our symmetry assumptions. Let M denote the $l \times k$ coefficient matrix represented by $M_{ij} = \sum_{J \in \mathsf{perm}(J_j)} \omega^{I_i \cdot J}$. To show that the system has a solution, it suffices to show that the column span of M is full. This is equivalent to showing that for each $i = 1, 2, \ldots l$ there exist coefficients $a_1, a_2, \ldots, a_k$ such that

$$\sum_{j=1}^{k} a_j \cdot \sum_{J \in \mathsf{perm}(J_j)} \omega^{I_i \cdot J} \neq 0$$

$$\sum_{j=1}^{k} a_j \cdot \sum_{J \in \mathsf{perm}(J_j)} \omega^{I_{i'} \cdot J} = 0 \ \forall i' \neq i$$

Fix an index i_0. We can view each equation above as a polynomial in $\omega^{[I_i]}$ given by

$$P(x_1, \ldots, x_n) = \sum_{j=1}^{k} a_j \sum_{J \in \mathsf{perm}(J_j)} x^J$$

and the problem becomes equivalent to constructing a polynomial that vanishes on $\omega^{[I_i]}$ if and only if $i \neq i_0$. Note that only the entries $x_{dm+1}, \ldots, x_n$ matter as we have $x_1 = \cdots = x_m = 1, \ldots, x_{(d-1)m+1} = \cdots = x_{dm} = \omega^{d-1}$ for all points we consider.

For $I_i = (i_1, i_2, \ldots i_n)$, let I_i' denote the sub-tuple $(i_{dm+1}, \ldots, i_n)$. The problem is equivalent to constructing a polynomial

$$Q(x_{dm+1}, \ldots, x_n) = P(1, 1, \ldots, \omega^{d-1}, \ldots, \omega^{d-1}, x_{dm+1}, \ldots x_n)$$

such that Q vanishes on $\omega^{[I_i']}$ if and only if $i \neq i_0$.

Lemma 27 implies that by choosing the coefficients $a_1, \ldots, a_k$, we can make Q be any polynomial that is symmetric in $x_{dm+1}, \ldots, x_n$ and degree at most $d-1$ in each of the variables.

Now consider the polynomial

$$Q_{i_0}(x_{dm+1}, \ldots, x_n) = \sum_{I' \in \mathsf{perm}(I_{i_0}')} \left(\frac{x_{dm+1}^d - 1}{x_{dm+1} - \omega^{I'0}}\right) \cdots \left(\frac{x_n^d - 1}{x_n - \omega^{I'(n-dm)}}\right)$$

(note this is a polynomial with coefficients in $\mathbb{C}$ since each of the factors reduces to a degree $d-1$ polynomial).

It is clear that the above polynomial is symmetric in all of the variables and satisfies the degree constraint so we know we can choose suitable coefficients $a_1, \ldots, a_k$. We claim that the polynomial we construct does not vanish on $\omega^{[I'_{i_0}]}$ but vanishes on $\omega^{[I'_i]}$ for $i \neq i_0$. Indeed, the product

$$\left(\frac{x_{dm+1}^d - 1}{x_{dm+1} - \omega^{I'0}}\right) \cdots \left(\frac{x_n^d - 1}{x_n - \omega^{I'(n-dm)}}\right)$$

is 0 if and only if $(x_{dm+1}, \ldots, x_n) \neq I'$. However, there is exactly one $I' \in \mathsf{perm}(I'_{i_0})$ with $I' = I'_{i_0}$ and none with $I' = I'_i$ for $i \neq i_0$ since $I_1, I_2, \ldots, I_l$ are representatives of distinct equivalence classes under permutation of entries. This means that the polynomial Q_{i_0} we constructed has the desired properties and completes the proof that the system is solvable. ◀

Proof of Lemma 28. Since $M(f) = (M(f) - M(f')) + M(f')$, to complete the proof of Lemma 28, it suffices to bound the number of nonzero entries in $M(f) - M(f')$ and the rank of $M(f')$.

The number of nonzero entries in each row and column of $(M(f) - M(f'))$ is at most $|T|$. This is exactly the number of elements of $\mathbb{Z}_d^n$ with at least $n\big(1-\delta\big)$ entries equal to 0. Using standard tail bounds on the binomial distribution, the probability of a random n-tuple having at least that many 0s is at most

$$e^{-nD(1-\delta||\frac{1}{d})} = e^{-n\left((1-\delta)\log(d(1-\delta))+\delta\log(\frac{d\delta}{d-1})\right)} = d^{-n(1-\delta)}e^{-n\left((1-\delta)\log(1-\delta)+\delta\log(\frac{d\delta}{d-1})\right)}$$

where $D(\cdot||\cdot)$ denotes KL-divergence. For $\delta < 0.01$, the above is at most $d^{-n(1-\sqrt{\delta})}$ and thus we change at most $d^{\epsilon n}$ entries in each row and column.

By Claim 23, the rank of $M(f')$ is at most $d^n - |\mathsf{perm}(S)|$. Equivalently, this is the number of n-tuples such that some element in $\{0, 1, \ldots, d-1\}$ appears less than $\big(\frac{1-\delta}{d}\big)n$ times. We use Hoeffding's inequality and then union bound over the d possibilites to get the probability that a randomly chosen n-tuple in $\mathbb{Z}_d^n$ is outside S is at most

$$de^{-2\frac{\delta^2 n}{d^2}} = e^{-2\frac{\delta^2 n}{d^2}+\log d}$$

When $n > \frac{d^2(\log d)^2}{\delta^2}$, the above is at most $d^{-\frac{\epsilon^4 n}{d^2 \log d}}$ and thus the rank of $M(f')$ is at most $d^{\left(1-\frac{\epsilon^4}{d^2\log d}\right)n}$, completing the proof of Lemma 28. ◀

Proof of Theorem 26. Applying Claim 25 and Lemma 28 we immediately get the desired. ◀

Using Theorem 26, Lemma 24, and Claim 21, we get the following result which extends Lemma 28 to matrices where f is not symmetric.

▶ **Corollary 30.** *For any function $f : \mathbb{Z}_d^n \to \mathbb{C}$ and any $0 < \epsilon < 0.1$ such that $n \geq \frac{d^2(\log d)^2}{\epsilon^4}$, we have*

$$r_{M(f)}(2N^{1-\frac{\epsilon^4}{d^2\log d}}) \leq N^{2\epsilon}$$

where $N = d^n$.

4 Non-rigidity for Fourier Matrices of Well-Factorable Size

Our goal in this section is to show that we can find infinitely many values of N for which the Fourier matrix F_N is highly non-rigid. The integers N we analyze will be products of many distinct primes q_i with the property that $q_i - 1$ is very smooth (has all prime factors small). For these values of N, we can decompose the matrix F_N into several submatrices that are closely related to Hadamard matrices. We then apply the results from the previous section to show that each submatrix is non-rigid and aggregate over the submatrices to conclude that F_N is non-rigid.

We first show precisely how to construct N. The properties that we want N to have are stated in the following two definitions.

▶ **Definition 31.** *We say a prime q is (α, x)-good if the following properties hold.*
- $x^{0.999} \leq q \leq x$
- *All prime powers dividing $q-1$ are at most x^α*

▶ **Definition 32.** *We say an integer N is (l, α, x)-factorable if the following properties hold.*
- $N = q_1 \dots q_l$ *where* $q_1, \dots, q_l$ *are distinct primes*
- $q_1, \dots, q_l$ *are all (α, x)-good*

To show the existence of (l, α, x)-factorable integers, it suffices to show that there are many (α, x)-good primes. This is captured in the following lemma.

▶ **Lemma 33.** *There exists a fixed constant C_0 such that for any parameter $\alpha > 0.2961$ and sufficiently large x (possibly depending on α), there are at least $\frac{x}{(\log x)^{C_0}}$ distinct (α, x)-good primes.*

The proof of Lemma 33 relies on the following result from analytic number theory, found in [2], that allows us to find a large set of primes q_i for which $q_i - 1$ is very smooth.

▶ **Definition 34.** *For a positive integer m, let $P^+(m)$ denote the largest prime factor of m. For a fixed positive integer a, let*

$$\pi_a(x, y) = |\{p | a < p \leq x, P^+(p-a) \leq y\}|$$

where p ranges over all primes. In other words, $\pi_a(x, y)$ is the number of primes at most x such that $p - a$ is y-smooth.

▶ **Theorem 35** ([2])**.** *There exist constants x_0, C such that for $\beta = 0.2961$, $x > x_0$ and $y \geq x^\beta$ we have* [5]

$$\pi_1(x, y) > \frac{x}{(\log x)^C}$$

Proof of Lemma 33. Let $y = x^\beta$ where $\beta = 0.2961$. By Theorem 35, for sufficiently large x, we can find at least $\lceil \frac{x}{(\log x)^C} - x^{0.999} \rceil$ primes $p_1, \dots, p_l$ between $x^{0.999}$ and x such that all prime factors of $p_i - 1$ are at most x^β. Eliminate all of the p_i such that one of the prime powers in the prime factorization of $p_i - 1$ is more than x^α. Note that there are at most x^β distinct primes that divide $p_i - 1$ for some i. Thus, there are at most $x^\beta \log x$ different prime powers bigger than x^α that divide some $p_i - 1$. Each of these prime powers can divide

[5] [2] proves the same inequality with $\pi_a(x, y)$ for any integer a where x_0 may depend on a and C is an absolute constant.

at most $x^{1-\alpha}$ of the elements $\{p_1, \ldots, p_l\}$, so in total, we eliminate at most $x^{1-\alpha+\beta} \log x$ of the p_i. Thus, for sufficiently large x, the number of (α, x)-good primes is at least

$$\frac{x}{(\log x)^C} - x^{0.999} - x^{1-\alpha+\beta} \log x \geq \frac{x}{2(\log x)^C}$$ ◀

For simplicity, we will set $\alpha = 0.3$ by default.

▶ **Definition 36.** *A prime is said to be x-good if it is $(0.3, x)$-good. An integer N is said to be (l, x)-factorable if it is $(l, 0.3, x)$-factorable.*

Lemma 33 implies that for all sufficiently large x and $l \leq \frac{x}{(\log x)^{C_0}}$, we can find (l, x)-factorable integers. We now show that if we choose x sufficiently large and N to be (l, x)-factorable for some $x^{0.99} \leq l \leq x^{0.993}$, then F_N is highly non-rigid.

▶ **Theorem 37.** *Let $0 < \epsilon < 0.1$ be given. For x sufficiently large and N a (l, x)-factorable number for $x^{0.99} \leq l \leq x^{0.993}$, we must have*

$$r_{F_N}\left(\frac{N}{2^{\epsilon^4 (\log N)^{0.0005}}}\right) \leq N^{4\epsilon}$$

In order to prove Theorem 37, we will first prove a series of preliminary results that characterize the structure of Fourier and Hadamard matrices.

4.1 Structure of Hadamard and Fourier Matrices

▶ **Lemma 38.** *Let $n = x_1 x_2 \ldots x_j$ for pairwise relatively prime positive integers $x_1, \ldots, x_j$. There exists a permutation of the rows and columns of F_n, say F' such that*

$$F' = F_{x_1} \otimes \cdots \otimes F_{x_j}$$

where $\otimes$ denotes the Kronecker product.

Proof. Let γ be a primitive n^{th} root of unity. For $i = 1, 2, \ldots j$, let $\gamma_i = \gamma^{c_i \frac{n}{x_i}}$ where c_i is chosen such that $c_i \frac{n}{x_i} \equiv 1 \mod x_i$. Note this is possible since $x_1, \ldots, x_j$ are pairwise relatively prime. γ_i is a primitive ${x_i}^{\text{th}}$ root of unity.

Now by the Chinese remainder theorem, there is a ring isomorphism between $\mathbb{Z}_n$ and $\mathbb{Z}_{x_1} \times \cdots \times \mathbb{Z}_{x_j}$. We can thus view F_n as a matrix whose rows and columns are indexed by elements of $\mathbb{Z}_{x_1} \times \cdots \times \mathbb{Z}_{x_j}$ and such that the entry $F_{n|AB}$ corresponding to tuples $A = (a_1, \ldots, a_j)$ and $B = (b_1, \ldots, b_j)$ is γ^c where c is the unique element of $\mathbb{Z}_n$ with $c \equiv a_i b_i \mod x_i$ for all i.

For each matrix F_{x_i} its rows and columns are labeled with elements of $\mathbb{Z}_{x_i}$ and its entries are $F_{x_i|ab} = \gamma_i^{a \cdot b}$. Thus in the Kronecker product, the rows and columns are labeled with elements of $\mathbb{Z}_{x_1} \times \cdots \times \mathbb{Z}_{x_j}$ such that the entry corresponding to tuples $(a_1, \ldots, a_j)$ and $(b_1, \ldots, b_j)$ is

$$\gamma_1^{a_1 b_1} \ldots \gamma_j^{a_j b_j} = \gamma^{c_1 a_1 b_1 \frac{n}{x_1} + \cdots + c_j a_j b_j \frac{n}{x_j}}$$

For each x_i, we compute the residue of the exponent in the above expression $\mod x_i$. The term $c_i a_i b_i \frac{n}{x_i}$ is congruent to $a_i b_i$ by definition and all other terms are 0 so the sum is congruent to $a_i b_i \mod x_i$. Thus, for some permutation of the rows and columns of F_n, it is equal to the Kronecker product $F_{x_1} \otimes \cdots \otimes F_{x_j}$, as desired. ◀

▶ **Lemma 39.** *Let $M = A \otimes B$ where A is an $m \times m$ matrix and B is an $n \times n$ matrix. For any two integers r_1, r_2 we have*

$$\mathsf{r}_M(r_1 n + r_2 m) \leq \mathsf{r}_A(r_1)\mathsf{r}_B(r_2)$$

Proof. The proof of this lemma is similar to the proof of Lemma 24. There are matrices E, F with atmost $\mathsf{r}_A(r_1)$ and $\mathsf{r}_B(r_2)$ nonzero entries respectively such that $\mathsf{rank}(A+E) \leq r_1$ and $\mathsf{rank}(B+F) \leq r_2$. We will now show that $\mathsf{rank}(M - E \otimes F) \leq r_1 n + r_2 m$. Indeed

$$M - E \otimes F = (A+E) \otimes B - E \otimes (B+F)$$

and the right hand side of the above has rank at most $r_1 n + r_2 m$ since rank multiplies under the Kronecker product. Clearly $E \otimes F$ has at most $\mathsf{r}_A(r_1)\mathsf{r}_B(r_2)$ nonzero entries in each row and column so we are done. ◀

▶ **Lemma 40.** *Consider the matrix*

$$A = (\underbrace{F_{t_1} \otimes \cdots \otimes F_{t_1}}_{a_1}) \otimes \cdots \otimes (\underbrace{F_{t_n} \otimes \cdots \otimes F_{t_n}}_{a_n})$$

Let $0 < \epsilon < 0.1$ be some chosen parameter. Assume $a_i \geq \frac{t_i^2 (\log t_i)^2}{\epsilon^4}$ for all i. Let $P = t_1^{a_1} \dots t_n^{a_n}$. Then

$$\mathsf{r}_A\left(P \sum_{i=1}^{n} \left(\frac{1}{t_i^{\frac{a_i \epsilon^4}{t_i^2 \log t_i}}} \right)\right) \leq P^{\epsilon}$$

Proof. Note $\underbrace{F_{t_1} \otimes \cdots \otimes F_{t_1}}_{a_1} = H_{t_1^{a_1}}$. Now we apply Theorem 26 to each of the n terms. Let $T_{t_i,a_i} = \underbrace{F_{t_i} \otimes \cdots \otimes F_{t_i}}_{a_i}$. We have

$$\mathsf{r}_{T_{t_i,a_i}}\left(t_i^{a_i\left(1 - \frac{\epsilon^4}{t_i^2 \log t_i}\right)} \right) \leq t_i^{a_i \epsilon}$$

Now we combine the above estimates over all i by repeatedly applying Lemma 39. We get

$$\mathsf{r}_A\left(\sum_{i=1}^{n} t_i^{a_i\left(1 - \frac{\epsilon^4}{t_i^2 \log t_i}\right)} \left(\frac{P}{t_i^{a_i}} \right) \right) \leq P^{\epsilon}$$

This easily rearranges into the desired. ◀

4.2 Proof of Theorem 37

To complete the proof of Theorem 37, we will break F_N into submatrices, show that each submatrix is non-rigid using techniques from the previous section, and then combine our estimates to conclude that F_N is non-rigid. Recall that N is (l, x)-factorable with $x^{0.99} \leq l \leq x^{0.993}$, meaning $N = q_1 q_2 \dots q_l$ for some distinct primes $q_1, \dots, q_l$ where $q_i - 1$ has no large prime power divisors for all i. Let γ be a primitive N^{th} root of unity.

▶ **Definition 41.** *For a subset $S \subset [l]$ define $\mathsf{mult}_N(S) = \prod_{s \in S} q_s$ and $\mathsf{fact}_N(S) = \prod_{s \in S}(q_s - 1)$.*

▶ **Definition 42.** *For all $S \subset [l]$ we will define T_S as the subset of $[N] \times [N]$ indexed by (i, j) such that*

$$ij \not\equiv 0 \mod q_s \ \forall s \in S$$
$$ij \equiv 0 \mod q_s \ \forall s \notin S$$

Note that as S ranges over all subsets of $[l]$, the sets T_S form a partition of $[N] \times [N]$.

For each S, we will divide the set T_S into submatrices such that when filled with the corresponding entries of F_N, we can apply Lemma 40 to show that each submatrix is nonrigid. The key intuition is that for a given prime q_i, once we restrict to nonzero residues, the multiplicative subgroup actually has the additive structure of $\mathbb{Z}_{q_i-1}$. Since $q_i - 1$ is smooth, $\mathbb{Z}_{q_i-1}$ is a direct sum of cyclic groups of small order.

▶ **Definition 43.** *For all $S \subset [l]$, we define the $\mathsf{fact}_N(S) \times \mathsf{fact}_N(S)$ matrix $M(S)$ as follows. Let R_S be the set of residues modulo $\mathsf{mult}_N(S)$ that are relatively prime to $\mathsf{mult}_N(S)$. Note that $|R_S| = \mathsf{fact}_N(S)$. Each row and each column of $M(S)$ is indexed by an element of R_S and the entry in row i and column j is $\theta^{i \cdot j}$ where θ is a primitive $\mathsf{mult}_N(S)$ root of unity. The exact order of the rows and columns will not matter for our uses. Note that replacing θ with θ^k for k relatively prime to $\mathsf{mult}_N(S)$ simply permutes the rows so it does not matter which root of unity we choose.*

▶ **Lemma 44.** *Consider the set of entries in F_N indexed by elements of T_S. We can partition this set into $\prod_{s \notin S}(2q_s - 1)$ submatrices each of size $\mathsf{fact}_N(S) \times \mathsf{fact}_N(S)$ that are equivalent to $M(S)$ up to some permutation of rows and columns.*

Proof. In T_S, for each prime q_s with $s \notin S$, there are $2q_s - 1$ choices for what i and j are mod q_s. Now fix the choice of $i, j \mod q_s$ for all $s \notin S$. Say we restrict to indices with $i \equiv c_1 \mod \prod_{s \notin S} q_s$ and $j \equiv c_2 \mod \prod_{s \notin S} q_s$.

We are left with a $\mathsf{fact}_N(S) \times \mathsf{fact}_N(S)$ matrix, call it A, where i and j run over all residues modulo $\mathsf{mult}_N(S)$ that are relatively prime to $\mathsf{mult}_N(S)$. Naturally, label all rows and columns of this matrix by what the corresponding indices i and j are modulo $\mathsf{mult}_N(S)$. For a row labeled a and a column labeled b, we compute the entry A_{ab}. The value is $\gamma^{a' \cdot b'}$ where a' is the unique element of $\mathbb{Z}_N$ such that $a' \equiv a \mod \mathsf{mult}_N(S)$ and $a' \equiv c_1 \mod \prod_{s \notin S} q_s$ and b' is defined similarly. We have

$$a' \cdot b' \equiv ab \mod \mathsf{mult}_N(S)$$

$$a' \cdot b' \equiv c_1 c_2 \equiv 0 \mod \prod_{s \notin S} q_s$$

Therefore

$$a'b' \equiv k \prod_{s \notin S} q_s ab \mod \mathsf{mult}_N(S)$$

where k is defined as an integer such that $k \prod_{s \notin S} q_s \equiv 1 \mod \mathsf{mult}_N(S)$. Note that k clearly exists since $\prod_{s \notin S} q_s$ and $\mathsf{mult}_N(S)$ are relatively prime. Since $\gamma^{k \prod_{s \notin S} q_s}$ is a primitive $\mathsf{mult}_N(S)$ root of unity, the matrix A is equivalent to $M(S)$ up to some permutation, as desired. ◀

▶ **Lemma 45.** *For a subset $S \subset [l]$ with $|S| = k$ and $M(S)$ (as defined in Definition 43) a $\mathsf{fact}_N(S) \times \mathsf{fact}_N(S)$ matrix as described above, we have*

$$r_{M(S)}\left(\frac{\mathsf{fact}_N(S)}{2^{\epsilon^4 x^{0.01}}}\right) \leq (\mathsf{fact}_N(S))^{3\epsilon}$$

as long as $k \geq x^{0.95}$

Proof. WLOG $S = \{1, 2, \ldots, k\}$. Consider the factorizations of $q_1 - 1, \ldots, q_k - 1$ into prime powers. For each prime power $p_i^{e_i} \leq x^{0.3}$, let $c(p_i^{e_i})$ be the number of indices j for which $p_i^{e_i}$ appears (exactly) in the factorization of $q_j - 1$. Note that

$$(q_1 - 1)\ldots(q_k - 1) = \prod_t t^{c(t)}$$

where t ranges over all prime powers at most $x^{0.3}$. Consider all prime powers $p_i^{e_i}$ for which $c(p_i^{e_i}) < x^{0.62}$.

$$\prod_{t, c(t) \leq x^{0.62}} t^{c(t)} \leq \left((x^{0.3})^{x^{0.62}}\right)^{x^{0.3}} \leq x^{x^{0.92}}$$

Now consider all prime powers say $t_1, \ldots, t_n$ for which $c(t_i) \geq x^{0.62}$. Let $P = t_1^{c(t_1)} \ldots t_n^{c(t_n)}$. From the above and the assumption that $k \geq x^{0.95}, q_i \geq x^{0.999}$, we know that

$$P \geq \frac{\mathsf{fact}_N(S)}{x^{x^{0.92}}} \geq (\mathsf{fact}_N(S))^{(1-\epsilon)}$$

We will use the prime powers t_i and Theorem 26 to show that $M(S)$ is not rigid. Note that we can associate each row and column of $M(S)$ to a k-tuple $(a_1, \ldots, a_k)$ where $a_i \in \mathbb{Z}_{q_i - 1}$ as follows. First, it is clear that each row and column of $M(S)$ can be associated to a k-tuple $(z_1, \ldots, z_k) \in \mathbb{Z}_{q_1}^* \times \cdots \times \mathbb{Z}_{q_k}^*$. Now $\mathbb{Z}_{q_i}^*$ can be viewed as a cyclic group on $q_i - 1$ elements. This allows us to create a bijection between the rows and columns of $M(S)$ and elements of $\mathbb{Z}_{q_1 - 1} \times \cdots \times \mathbb{Z}_{q_k - 1}$.

Also note that for a row indexed by $A = (a_1, \ldots, a_k)$ and a column indexed by $B = (b_1, \ldots, b_k)$, the entry $M(S)_{AB}$ is dependent only on $A + B$. We will now decompose $M(S)$ into several $P \times P$ submatrices. In particular, we can write $q_i - 1 = d_i T_i$ where T_i is a product of some subset of $\{t_1, \ldots, t_n\}$ and d_i is relatively prime to T_i. We have $T_1 T_2 \ldots T_k = P$. For each $A', B' \in \mathbb{Z}_{d_1} \times \cdots \times \mathbb{Z}_{d_k}$, we can construct a $P \times P$ submatrix $M(S, A', B')$ consisting of all entries $M(S)_{AB}$ of $M(S)$ such that $A \equiv A', B \equiv B'$ (where the equivalence is over $\mathbb{Z}_{d_1} \times \cdots \times \mathbb{Z}_{d_k}$). This gives us d^2 different submatrices where $d = d_1 \ldots d_k$. Naturally, we can associate each row and column of a submatrix $M(S, A', B')$ with an element of $\mathbb{Z}_{T_1} \times \cdots \times \mathbb{Z}_{T_k}$ such that for a row labeled I and a column labeled J, the entry $M(S, A', B')_{IJ}$ only depends on $I + J$. In particular, this means that $X\,(M(S, A', B'))\,X$ is diagonal where $X = F_{T_1} \otimes \cdots \otimes F_{T_k}$. Now, using Lemma 38, we can rewrite

$$X = (\underbrace{F_{t_1} \otimes \cdots \otimes F_{t_1}}_{c(t_1)}) \otimes \cdots \otimes (\underbrace{F_{t_n} \otimes \cdots \otimes F_{t_n}}_{c(t_n)})$$

Since for x sufficiently large, $c(t_i) \geq x^{0.62} \geq \frac{t_i^2 (\log t_i)^2}{\epsilon^4}$, we can use Lemma 40 and get that

$$\mathsf{r}_X\left(P \sum_{i=1}^{n} \left(\frac{1}{t_i^{\frac{c(t_i)\epsilon^4}{t_i^2 \log t_i}}}\right)\right) \leq P^{\epsilon}$$

Let E be the matrix of changes to reduce the rank of X according to the above. We have that E has at most P^ϵ nonzero entries in each row and column and

$$\mathsf{rank}(X-E) \leq P\sum_{i=1}^{n}\left(\frac{1}{t_i^{\frac{c(t_i)\epsilon^4}{t_i^2\log t_i}}}\right)$$

We can write $M(S)$ in block form as

$$\begin{bmatrix} M(S,A_1,B_1) & M(S,A_1,B_2) & \dots & M(S,A_1,B_d) \\ M(S,A_2,B_1) & M(S,A_2,B_2) & \dots & M(S,A_1,B_d) \\ \vdots & \vdots & \ddots & \vdots \\ M(S,A_d,B_1) & M(S,A_d,B_2) & \dots & M(S,A_d,B_d) \end{bmatrix}$$

where $A_1,\dots,A_d$ and $B_1,\dots,B_d$ range over the elements of $\mathbb{Z}_{d_1}\times\cdots\times\mathbb{Z}_{d_k}$. We can rearrange the above as

$$\begin{bmatrix} M(S,A_1,B_1) & \dots & M(S,A_1,B_d) \\ \vdots & \ddots & \vdots \\ M(S,A_d,B_1) & \dots & M(S,A_d,B_d) \end{bmatrix} = \begin{bmatrix} XD_{11}X & \dots & XD_{1d}X \\ \vdots & \ddots & \vdots \\ XD_{d1}X & \dots & XD_{dd}X \end{bmatrix}$$

where the D_{ij} are diagonal matrices. Now consider the matrix

$$E(S) = \begin{bmatrix} ED_{11}E & \dots & ED_{1d}E \\ \vdots & \ddots & \vdots \\ ED_{d1}E & \dots & ED_{dd}E \end{bmatrix}$$

We have

$$M(S)-E(S) = \begin{bmatrix} XD_{11}X-ED_{11}E & \dots & XD_{1d}X-ED_{1d}E \\ \vdots & \ddots & \vdots \\ XD_{d1}X-ED_{d1}E & \dots & XD_{dd}X-ED_{dd}E \end{bmatrix} =$$

$$\begin{bmatrix} XD_{11}(X-E) & \dots & XD_{1d}(X-E) \\ \vdots & \ddots & \vdots \\ XD_{d1}(X-E) & \dots & XD_{dd}(X-E) \end{bmatrix} + \begin{bmatrix} (X-E)D_{11}E & \dots & (X-E)D_{1d}E \\ \vdots & \ddots & \vdots \\ (X-E)D_{d1}E & \dots & (X-E)D_{dd}E \end{bmatrix}$$

In the above expression, each of the two terms has rank at most

$$dP\sum_{i=1}^{n}\left(\frac{1}{t_i^{\frac{c(t_i)\epsilon^4}{t_i^2\log t_i}}}\right) = \mathsf{fact}_N(S)\sum_{i=1}^{n}\left(\frac{1}{t_i^{\frac{c(t_i)\epsilon^4}{t_i^2\log t_i}}}\right) \leq \frac{1}{2}\left(\frac{\mathsf{fact}_N(S)}{2^{\epsilon^4 x^{0.01}}}\right)$$

Note that when computing the rank, we only multiply by d (and not d^2) because the small blocks are all multiplied by the same low rank matrix on either the left or right. The number of nonzero entries in each row and column of $E(S)$ is at most $P^{2\epsilon}d = \frac{\mathsf{fact}_N(S)}{P^{1-2\epsilon}}$. Since $P \geq (\mathsf{fact}_N(S))^{1-\epsilon}$, we conclude

$$\mathsf{r}_{M(S)}\left(\frac{\mathsf{fact}_N(S)}{2^{\epsilon^4 x^{0.01}}}\right) \leq (\mathsf{fact}_N(S))^{3\epsilon}$$ ◀

We are now ready to complete the analysis of the non-rigidity of the Fourier transform matrix F_N.

Proof of Theorem 37. Set the threshold $k_0 = l\left(1 - \frac{\epsilon^4}{x^{0.985}}\right)$. The sets T_S, as S ranges over all subsets of $[l]$, form a partition of $[N] \times [N]$. For each $S \subset [l]$ with $|S| \geq k_0$, we will divide T_S into $\mathsf{fact}_N(S) \times \mathsf{fact}_N(S)$ submatrices using Lemma 44 and change entries to reduce the rank of every submatrix according to Lemma 45. We will not touch the entries in sets T_S for $|S| < k_0$. Call the resulting matrix M'. We now estimate the rank of M' and then the maximum number of entries changed in any row or column.

Let $m = l - k_0 = \frac{\epsilon^4}{x^{0.985}} l$. Note since $l \geq x^{0.99}$, $m \geq \epsilon^4 x^{0.005}$. We remove all rows and columns corresponding to integers divisible by at least $\frac{m}{2}$ of the primes $q_1, \ldots, q_l$. Since $l \leq x^{0.993}$. The number of rows and columns removed is at most

$$N\left(\sum_{S\subset[l],|S|=\frac{m}{2}} \prod_{i\in S} \frac{1}{q_i}\right) \leq \frac{N}{x^{0.999\frac{m}{2}}}\binom{l}{\frac{m}{2}} < \frac{N}{x^{0.999\frac{m}{2}}} l^{\frac{m}{2}} \leq \frac{N}{x^{0.003m}}$$

The remaining entries must be subdivided into matrices of the form $M(S)$ for various subsets $S \subset [l]$, $|S| \geq k_0$. Say $q_1 < q_2 < \cdots < q_l$. The number of such submatrices is at most

$$\frac{N^2}{((q_1-1)\ldots(q_{k_0}-1))^2} \leq (q_{k_0+1}\ldots q_l)^2 \left(\frac{q_1\ldots q_{k_0}}{(q_1-1)\ldots(q_{k_0}-1)}\right)^2 \leq 3(q_{k_0+1}\ldots q_l)^2 \leq 3x^{2m}$$

Each one of the submatrices has rank at most

$$\frac{N}{2^{\epsilon^4 x^{0.01}}}$$

so in total the rank is at most

$$N\frac{3x^{2m}}{2^{\epsilon^4 x^{0.01}}} \leq \frac{N}{2^{\epsilon^4 x^{0.002}}}$$

Combining the two parts we easily get

$$\mathsf{rank}(M') \leq \frac{N}{2^{\epsilon^4 x^{0.001}}}$$

Now we bound the number of entries changed. The number of entries changed in each row or column is at most

$$\frac{N}{((q_1-1)\ldots(q_{k_0}-1))} N^{3\epsilon} \leq (q_{k_0+1}\ldots q_l)\left(\frac{q_1\ldots q_{k_0}}{(q_1-1)\ldots(q_{k_0}-1)}\right) N^{3\epsilon} \leq 3N^{3\epsilon+1.1\frac{m}{l}} \leq N^{4\epsilon}$$

As $2^{\epsilon^4 x^{0.001}} \geq 2^{\epsilon^4 (\log N)^{0.0005}}$ for sufficiently large x, we conclude

$$\mathsf{r}_{F_N}\left(\frac{N}{2^{\epsilon^4(\log N)^{0.0005}}}\right) \leq N^{4\epsilon}$$ ◀

5 Non-rigidity of All Fourier Matrices

In the previous section, we showed that there exists an infinite set of Fourier matrices that are not Valiant-rigid. In this section, we will bootstrap the results from Section 4 to show that in fact, all sufficiently large Fourier matrices are not rigid.

The first ingredient will be a stronger form of Lemma 33. Recall that a prime q is defined to be x-good if $x^{0.999} \leq q \leq x$ and all prime powers dividing $q-1$ are at most $x^{0.3}$ and that an integer N is defined to be (l, x)-factorable if it can be written as the product of l distinct x-good primes.

▶ **Lemma 46.** *For all sufficiently large integers K, there exist l, x, N such that $x^{0.99} \leq l \leq x^{0.993}$, N is (l, x)-factorable, and $K < N < K(\log K)^2$.*

Proof. Call an N well-factorable if it is (l, x)-factorable for some x and $x^{0.99} \leq l \leq x^{0.993}$. Let N_0 be the largest integer that is well-factorable with $N_0 \leq K$. Say N_0 is (l, x)-factorable.

We have $N_0 = q_1 \dots q_l$ where $q_1, \dots, q_l$ are distinct, x-good primes. If $l < \lfloor x^{0.993} \rfloor$ then by Lemma 33, we can find another x-good prime q_{l+1}. We can then replace N_0 with $q_{l+1}N_0$. $q_{l+1}N_0 > K$ by the maximality of N_0 and also $q_{l+1}N_0 \leq N_0 x \leq N_0(\log N_0)^2$ so $q_{l+1}N_0$ satisfies the desired properties.

We now consider the case where $l = \lfloor x^{0.993} \rfloor$. First, if $q_1, \dots, q_l$ are not the l largest x-good primes then we can replace one of them say q_1 with $q_1' > q_1$. The number $N' = q_1' q_2 \dots q_l$ is well-factorable and between N_0 and $N_0 x^{0.001}$. Using the maximality of N_0, we deduce that N' must be in the desired range.

On the other hand if $q_1, \dots, q_l$ are the l largest x-good primes, we know they are actually all between $x^{0.9995}$ and x. This is because by Lemma 33, there are more than $x^{0.9995} + l$ distinct x-good primes. Let C be the constant in Theorem 35 and let $x' = x(\log x)^{C_0+1}$. The above implies that $q_1, \dots, q_l$ are x'-good and clearly $x'^{0.99} \leq l < x'^{0.993}$. By Lemma 33, there are more than x distinct x'-good primes so there exists some $q > x$ that is x'-good. The product $N' = q_1 \dots q_{l-1} q$ is well-factorable and larger than N_0 so $N' > K$. Also $N' \leq N_0 x^{0.001}(\log x)^{C_0+1} < K(\log K)^2$ so N' is in the desired range. ◀

Also note that as a consequence of Theorem 37 we have:

▶ **Lemma 47.** *Let $0 < \epsilon < 0.1$ be fixed, x sufficiently large, and N_0 a (l, x)-factorable integer with $x^{0.99} \leq l \leq x^{0.993}$. If $\frac{N_0}{(\log N_0)^2} \leq N \leq \frac{N_0}{2}$ then any $N \times N$ adjusted-circulant matrix M satisfies*

$$r_M\left(\frac{N}{2^{\epsilon^4(\log N)^{0.0004}}}\right) \leq N^{9\epsilon}$$

Proof. By Claim 22, Lemma 24 and Theorem 37, any adjusted-circulant matrix M_0 of size N_0 satisfies

$$\mathsf{r}_{M_0}\left(\frac{2N_0}{2^{\epsilon^4(\log N_0)^{0.0005}}}\right) \leq N_0^{8\epsilon}$$

Any adjusted-circulant matrix of size at most $\frac{N_0}{2}$ can be embedded (in the upper left corner) of an adjusted-circulant matrix of size N_0 so we have the same inequality for the matrix M. Rewriting the bounds in terms of N, we get the desired. ◀

We now have all of the parts to prove that all Fourier matrices are highly non-rigid.

▶ **Theorem 48.** *For any fixed $0 < \epsilon < 0.1$ and N sufficiently large,*

$$r_{F_N}\left(\frac{N}{2^{\epsilon^4(\log N)^{0.0004}}}\right) \leq N^{9\epsilon}$$

Proof. By Lemma 46, we can find an integer N_0 such that $2N < N_0 < 2N(\log 2N)^2$ and N_0 is (l, x) factorable for some x and $x^{0.99} \leq l \leq x^{0.993}$. We have $\frac{N_0}{(\log N_0)^2} \leq N \leq \frac{N_0}{2}$. Thus, by Lemma 47, all $N \times N$ adjusted-circulant matrices satisfy.

$$\mathsf{r}_M\left(\frac{N}{2^{\epsilon^4(\log N)^{0.0004}}}\right) \leq N^{9\epsilon}$$

Note that by Claim 25, the rows and columns of F_N can be rescaled to obtain an adjusted-circulant matrix so F_N also satisfies the above inequality, completing the proof. ◀

We can now conclude that all adjusted-circulant and Hankel matrices are not Valiant-rigid.

▶ **Corollary 49.** *Let $0 < \epsilon < 0.1$ be fixed. For all sufficiently large N, if M is an $N \times N$ adjusted-circulant (or Hankel) matrix*

$$r_M\left(\frac{N}{2^{\epsilon^4(\log N)^{0.0003}}}\right) \leq N^{20\epsilon}$$

Proof. The result for circulant matrices follows immediately from the above and Lemma 24. For Hankel matrices, note that it is possible to embed any Hankel matrix of size N into the top left corner of a circulant matrix of size $2N$. ◀

6 Non-rigidity of Group Algebra Matrices for Abelian Groups

Using the results from the previous section, we can show that group algebra matrices for any abelian group are not Valiant-rigid.

▶ **Theorem 50.** *Let $0 < \epsilon < 0.1$ be fixed. Let G be an abelian group and $f : G \to \mathbb{C}$ be a function. Let $M = M_G(f)$ be the adjusted group algebra matrix. If $|G|$ is sufficiently large then*

$$r_M\left(\frac{2|G|}{2^{\epsilon^6(\log |G|)^{0.0001}}}\right) \leq |G|^{26\epsilon}$$

Proof. By the fundamental theorem of finite abelian groups, we can write $G = \mathbb{Z}_{n_1} \oplus \cdots \oplus \mathbb{Z}_{n_a}$. In light of Lemma 24, it suffices to bound the rigidity of $F = F_{n_1} \otimes \cdots \otimes F_{n_a}$.

WLOG, $n_1 \leq n_2 \leq \cdots \leq n_a$. We will choose k to be a fixed, sufficiently large positive integer. By Theorem 48, we can ensure that for $N > k$

$$\mathsf{r}_{F_N}\left(\frac{N}{2^{\epsilon^4(\log N)^{0.0004}}}\right) \leq N^{9\epsilon}$$

Consider the ranges $I_1 = [k, k^2), I_2 = [k^2, k^4), \ldots I_j = [k^{2^{j-1}}, k^{2^j}) \ldots$ and so on. Let S_j be a multiset defined by $S_j = I_j \cap \{n_1, \ldots, n_a\}$. Fix a j and say the elements of S_j are $x_1 \leq \cdots \leq x_b$. By Theorem 48, for each x_i, there are matrices E_{x_i} and A_{x_i} such that $F_{x_i} = A_{x_i} + E_{x_i}$, E_{x_i} has at most $x_i^{9\epsilon}$ nonzero entries in each row and column, and

$$\mathsf{rank}(A_{x_i}) \leq \frac{x_i}{2^{\epsilon^4(\log x_i)^{0.0004}}}$$

Now we can write

$$M_j = F_{x_1} \otimes \cdots \otimes F_{x_b} = (A_{x_1} + E_{x_1}) \otimes \cdots \otimes (A_{x_b} + E_{x_b}) = \sum_{S \subset [b]} \left(\bigotimes_{i \in S} A_{x_i}\right) \otimes \left(\bigotimes_{i' \notin S} E_{x'_i}\right)$$

$$= \sum_{S \subset [b], |S| \geq \epsilon b} \left(\bigotimes_{i \in S} A_{x_i}\right) \otimes \left(\bigotimes_{i' \notin S} E_{x'_i}\right) + \sum_{S \subset [b], |S| < \epsilon b} \left(\bigotimes_{i \in S} A_{x_i}\right) \otimes \left(\bigotimes_{i' \notin S} E_{x'_i}\right)$$

Let the first term above be N_1 and the second term be N_2. We will bound the rank of N_1 and the number of nonzero entries in each row and column of N_2. Note that by grouping the terms in the sum for N_1 we can write it in the form

$$\sum_{S \subset [b], |S| = \lceil \epsilon b \rceil} \bigotimes_{i \in S} A_{x_i} \otimes E_S$$

where for each S, E_S is some matrix. This implies that

$$\begin{aligned}\mathsf{rank}(N_1) &\le \binom{b}{\lceil \epsilon b\rceil}\frac{x_1\dots x_b}{\left(2^{\epsilon^4(\log x_1)^{0.0004}}\right)^{\lceil \epsilon b\rceil}}\\ &\le \frac{b^{\lceil \epsilon b\rceil}}{\left(\frac{\epsilon b}{3}\right)^{\lceil \epsilon b\rceil}}\frac{x_1\dots x_b}{\left(2^{\epsilon^4(\log x_1)^{0.0004}}\right)^{\lceil \epsilon b\rceil}} = x_1\dots x_b\left(\frac{3}{\epsilon 2^{\epsilon^4(\log x_1)^{0.0004}}}\right)^{\lceil \epsilon b\rceil}\end{aligned}$$

As long as k is sufficiently large, we have

$$\begin{aligned}\mathsf{rank}(N_1) &\le x_1\dots x_b\left(\frac{3}{\epsilon 2^{\epsilon^4(\log x_1)^{0.0004}}}\right)^{\lceil \epsilon b\rceil}\\ &\le x_1\dots x_b\left(\frac{1}{2^{\epsilon^4(\log x_1)^{0.0003}}}\right)^{\lceil \epsilon b\rceil} \le \frac{x_1\dots x_b}{2^{\epsilon^5(\log x_1\dots x_b)^{0.0002}}}\end{aligned}$$

where in the last step we used the fact that $x_i \le x_1^2$ for all i. The number of nonzero entries in each row or column of N_2 is at most

$$2^b x_b\dots x_{b-\lfloor \epsilon b\rfloor+1}(x_{b-\lfloor \epsilon b\rfloor}\dots x_1)^{9\epsilon} = 2^b(x_1\dots x_b)^{9\epsilon}(x_b\dots x_{b-\lfloor \epsilon b\rfloor+1})^{1-9\epsilon} \le (x_1\dots x_b)^{12\epsilon}$$

Note in the last step above, we used the fact that $x_i \le x_1^2$.

For each integer c between 2 and k, let n_c be the number of copies of c in the set $\{n_1,\dots,n_a\}$. If $n_c \ge \frac{k^2(\log k)^2}{\epsilon^4}$ then by Theorem 26, if we define $A_c = \underbrace{F_c\otimes\dots\otimes F_c}_{n_c}$ then

$$\mathsf{r}_{A_c}\left(c^{n_c(1-\frac{\epsilon^4}{k^2\log k})}\right) \le c^{n_c\epsilon}$$

Let $L = \lceil 2\log\log|G|\rceil$ and ensure that $|G|$ is sufficiently large so that $L > k$. Let T be the set of integers c between 2 and k such that $c^{n_c} \ge |G|^{\frac{\epsilon}{2L}}$ (note that as long as $|G|$ is sufficiently large, all elements of T must satisfy $n_c \ge \frac{k^2(\log k)^2}{\epsilon^4}$). Let R be the set of indices j for which $\prod_{x\in S_j} x \ge |G|^{\frac{\epsilon}{2L}}$. Since S_j is clearly empty for $j \ge L$, the matrix F can be written as

$$F = \left(\bigotimes_{2\le c<k}\left(\underbrace{F_c\otimes\dots\otimes F_c}_{n_c}\right)\right)\otimes\left(\bigotimes_{1\le j\le L} M_j\right)$$

Define

$$B = \left(\bigotimes_{c\notin T}\left(\underbrace{F_c\otimes\dots\otimes F_c}_{n_c}\right)\right)\otimes\left(\bigotimes_{j\notin R} M_j\right)$$

Note that the size of B is at most $\left(|G|^{\frac{\epsilon}{2L}}\right)^{k+L} \le |G|^{\epsilon}$. Also $F = B\otimes D$ where

$$D = \left(\bigotimes_{c\in T}\left(\underbrace{F_c\otimes\dots\otimes F_c}_{n_c}\right)\right)\otimes\left(\bigotimes_{j\in R} M_j\right)$$

For any rank r, $\mathsf{r}_M(|B|r) \le |B|\mathsf{r}_D(r)$. Applying Lemma 39 iteratively, we get

$$\mathsf{r}_D\left(\frac{|G|}{|B|}\left(\sum_{c\in T}\frac{1}{c^{n_c\frac{\epsilon^4}{k^2\log k}}} + \sum_{j\in R}\frac{1}{2^{\epsilon^5(\log\prod_{x\in S_j}x)^{0.0002}}}\right)\right) \le \left(\frac{|G|}{|B|}\right)^{12\epsilon}$$

Note that

$$\left(\sum_{c\in T}\frac{1}{c^{n_c\frac{\epsilon^4}{k^2\log k}}}+\sum_{j\in R}\frac{1}{2^{\epsilon^5(\log\prod_{x\in S_j}x)^{0.0002}}}\right)\leq\frac{k}{|G|^{\frac{\epsilon^5}{2Lk^2\log k}}}+\frac{L}{2^{\epsilon^6\left(\frac{\log|G|}{2L}\right)^{0.0002}}}\leq\frac{1}{2^{\epsilon^6(\log|G|)^{0.0001}}}$$

Overall, we conclude

$$\mathsf{r}_F\left(\frac{|G|}{2^{\epsilon^6(\log|G|)^{0.0001}}}\right)\leq|B|\left(\frac{|G|}{|B|}\right)^{12\epsilon}\leq|G|^{13\epsilon}$$

Since FMF is diagonal, Lemma 24 gives the desired. ◀

7 Final Remarks

7.1 Rigidity over Fields and Extensions

The proofs in the previous sections actually tell us slightly more about the non-rigidity of Fourier and circulant matrices than what is stated in our results. Firstly, our proof of Theorem 48 easily generalizes to any field where the necessary roots of unity (for the Fourier matrix and the generalized Hadamard matrices we embed into it) exist. Over a finite field of characteristic p, it is not difficult to adapt our proof in order to avoid using roots of unity of order p (as $x^p-1=(x-1)^p$). Also note that in our proof of non-rigidity for F_N (the $N\times N$ Fourier matrix), the changes we make to the entries all live in a number field of dimension polynomial in N. Combining this insight with Lemma 24 gives us:

▶ **Corollary 51.** *Let $0<\epsilon<0.1$ be fixed. Let M be an $N\times N$ circulant matrix with entries in a field $\mathbb{F}$. For N sufficiently large, there exists an algebraic extension of $\mathbb{F}$ with dimension polynomial in N, say $\mathbb{E}$, such that over $\mathbb{E}$*

$$r_M\left(\frac{N}{2^{\epsilon^4(\log N)^{0.0003}}}\right)\leq N^{20\epsilon}$$

In particular, M is not rigid over the algebraic closure of $\mathbb{F}$.

In fact, we get a slightly stronger result, that for any circulant matrix M, the locations of the entries that need to be changed is fixed and the changes are fixed linear combinations of the entries. This is important because when following Valiant's graph-theoretic approach for proving circuit lower bounds in [12], this slightly weaker notion of rigidity is exactly what is necessary to prove lower bounds. Our result is thus a strong indication that Valiant's overall graph-theoretic approach cannot be used to prove circuit lower bounds for computing convolutions (which correspond to circulant matrices).

Corollary 51 naturally raises the question of whether matrices can be rigid over some small field $\mathbb{F}$ but non-rigid over some extension. While it seems unlikely to expect the rigidity over any field $\mathbb{F}$ to equal the rigidity over any extension, we think it is an interesting open question to consider when it might be possible to relate (asymptotically) the rigidity of a family of matrices over $\mathbb{F}$ to their rigidity over various extensions of $\mathbb{F}$.

7.2 Group Algebra Matrices

Theorem 50 naturally raises the question of what happens when G is a non-abelian group. When G is non-abelian, it is no longer possible to diagonalize the matrix $M_G(f)$ but there is a change of basis matrix A such that $AM_G(f)A^*$ is block-diagonal where the diagonal blocks

correspond to the irreducible representations of G. When all of the irreducible representations of G are small, it may be possible to use similar techniques to the ones used here. On the other hand, this suggests that perhaps $M_G(f)$ is a candidate for rigidity when all irreducible representations of G are large (for instance quasi-random groups [6]).

References

1. Josh Alman and Ryan Williams. Probabilistic Rank and Matrix Rigidity. *CoRR*, abs/1611.05558, 2016. `arXiv:1611.05558`.
2. R. Baker and G. Harman. Shifted primes without large prime factors. *Acta Arithmetica*, 83(4):331–361, 1998.
3. Zeev Dvir and Benjamin Edelman. Matrix rigidity and the Croot-Lev-Pach lemma. *arXiv preprint*, 2017. `arXiv:1708.01646`.
4. Joel Friedman. A note on matrix rigidity. *Combinatorica*, 13(2):235–239, 1993.
5. Oded Goldreich and Avishay Tal. Matrix rigidity of random toeplitz matrices. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 91–104. ACM, 2016.
6. William T Gowers. Quasirandom groups. *Combinatorics, Probability and Computing*, 17(3):363–387, 2008.
7. Abhinav Kumar, Satyanarayana V Lokam, Vijay M Patankar, and MN Jayalal Sarma. Using elimination theory to construct rigid matrices. *computational complexity*, 23(4):531–563, 2014.
8. Satyanarayana V Lokam. On the rigidity of Vandermonde matrices. *Theoretical Computer Science*, 237(1-2):477–483, 2000.
9. Satyanarayana V Lokam. Quadratic lower bounds on matrix rigidity. In *International Conference on Theory and Applications of Models of Computation*, pages 295–307. Springer, 2006.
10. Satyanarayana V Lokam et al. Complexity lower bounds using linear algebra. *Foundations and Trends® in Theoretical Computer Science*, 4(1–2):1–155, 2009.
11. Mohammad Amin Shokrollahi, Daniel A Spielman, and Volker Stemann. A remark on matrix rigidity. *Information Processing Letters*, 64(6):283–285, 1997.
12. Leslie G. Valiant. Graph-theoretic arguments in low-level complexity. In Jozef Gruska, editor, *Mathematical Foundations of Computer Science 1977*, pages 162–176, Berlin, Heidelberg, 1977. Springer Berlin Heidelberg.

Nullstellensatz Size-Degree Trade-offs from Reversible Pebbling

Susanna F. de Rezende
KTH Royal Institute of Technology, Stockholm, Sweden

Jakob Nordström
University of Copenhagen, Denmark
KTH Royal Institute of Technology, Stockholm, Sweden

Or Meir
University of Haifa, Israel
http://cs.haifa.ac.il/~ormeir/
ormeir@cs.haifa.ac.il

Robert Robere
DIMACS, New Brunswick, U.S.A.

Abstract

We establish an exactly tight relation between reversible pebblings of graphs and Nullstellensatz refutations of pebbling formulas, showing that a graph G can be reversibly pebbled in time t and space s if and only if there is a Nullstellensatz refutation of the pebbling formula over G in size $t+1$ and degree s (independently of the field in which the Nullstellensatz refutation is made). We use this correspondence to prove a number of strong size-degree trade-offs for Nullstellensatz, which to the best of our knowledge are the first such results for this proof system.

2012 ACM Subject Classification Theory of computation → Proof complexity

Keywords and phrases proof complexity, Nullstellensatz, pebble games, trade-offs, size, degree

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.18

Funding This work was mostly carried out while the authors were visiting the Simons Institute for the Theory of Computing in association with the DIMACS/Simons Collaboration on Lower Bounds in Computational Complexity, which is conducted with support from the National Science Foundation.
Susanna F. de Rezende: was supported by the *Knut and Alice Wallenberg* grant KAW 2016.0066 *Approximation and Proof Complexity*.
Jakob Nordström: was supported by the *Knut and Alice Wallenberg* grant KAW 2016.0066 *Approximation and Proof Complexity* and by the Swedish Research Council grants 621-2012-5645 and 2016-00782.
Or Meir: was supported by the Israel Science Foundation (grant No. 1445/16).
Robert Robere: was supported by NSERC, and also conducted part of this work at DIMACS with support from the National Science Foundation under grant number CCF-1445755.

Acknowledgements We are grateful for many interesting discussions about matters pebbling-related (and not-so-pebbling-related) with Arkadev Chattopadhyay, Toniann Pitassi, and Marc Vinyals.

1 Introduction

In this work, we obtain strong trade-offs in proof complexity by making a connection to pebble games played on graphs. In this introductory section we start with a brief overview of these two areas and then explain how our results follow from connecting the two.

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka; Article No. 18; pp. 18:1–18:16

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

1.1 Proof Complexity

Proof complexity is the study of efficiently verifiable certificates for mathematical statements. More concretely, statements of interest claim to provide correct answers to questions like:

- Given a CNF formula, does it have a satisfying assignment or not?
- Given a set of polynomials over some finite field, do they have a common root?

There is a clear asymmetry here in that it seems obvious what an easily verifiable certificate for positive answers to the above questions should be, while it is not so easy to see what a concise certificate for a negative answer could look like. The focus of proof complexity is therefore on the latter scenario.

In this paper we study the algebraic proof system system *Nullstellensatz* introduced by Beame et al. [7]. A *Nullstellensatz refutation* of a set of polynomials $\mathcal{P} = \{p_i \mid i \in [m]\}$ with coefficients in a field $\mathbb{F}$ is an expression

$$\sum_{i=1}^{m} r_i \cdot p_i + \sum_{j=1}^{n} s_j \cdot (x_j^2 - x_j) = 1 \tag{1}$$

(where r_i, s_j are also polynomials), showing that 1 lies in the polynomial ideal in the ring $\mathbb{F}[x_1, \ldots, x_n]$ generated by $\mathcal{P} \cup \left\{x_j^2 - x_j \,\middle|\, j \in [n]\right\}$. By (a slight extension of) Hilbert's Nullstellensatz, such a refutation exists if and only if there is no common $\{0,1\}$-valued root for the set of polynomials $\mathcal{P}$.

Nullstellensatz can also be viewed as a proof system for certifying the unsatisfiability of CNF formulas. If we translate a clause like, e.g., $C = x \vee y \vee \overline{z}$ to the polynomial $p(C) = (1-x)(1-y)z = z - yz - xz + xyz$, then an assignment to the variables in a CNF formula $F = \bigwedge_{i=1}^{m} C_i$ (where we think of 1 as true and 0 as false) is satisfying precisely if all the polynomials $\{p(C_i) \mid i \in [m]\}$ vanish.

The *size* of a Nullstellensatz refutation (1) is the total number of monomials in all the polynomials $r_i \cdot p_i$ and $s_j \cdot (x_j^2 - x_j)$ expanded out as linear combinations of monomials. Another, more well-studied, complexity measure for Nullstellensatz is *degree*, which is defined as $\max\{\deg(r_i \cdot p_i), \deg(s_j \cdot (x_j^2 - x_j))\}$.

In order to prove a lower bound d on the Nullstellensatz degree of refuting a set of polynomials $\mathcal{P}$, one can construct a *d-design*, which is a map D from degree-d polynomials in $\mathbb{F}[x_1, \ldots, x_n]$ to $\mathbb{F}$ such that

1. D is linear, i.e., $D(\alpha p + \beta q) = \alpha D(p) + \beta D(q)$ for $\alpha, \beta \in \mathbb{F}$;
2. $D(1) = 1$;
3. $D(rp) = 0$ for all $p \in \mathcal{P}$ and $r \in \mathbb{F}[x_1, \ldots, x_n]$ such that $\deg(rp) \leq d$;
4. $D(x^2 s) = D(xs)$ for all $s \in \mathbb{F}[x_1, \ldots, x_n]$ such that $\deg(s) \leq d - 2$.

Designs provide a characterization of Nullstellensatz degree in that there is a d-design for $\mathcal{P}$ if and only if there is no Nullstellensatz refutation of $\mathcal{P}$ in degree d [18]. Another possible approach to prove degree lower bounds is by computationally efficient versions of Craig's interpolation theorem. It was shown in [53] that constant-degree Nullstellensatz refutations yield polynomial-size monotone span programs, and that this is also tight: every span program is a unique interpolant for some set of polynomials refutable by Nullstellensatz. This connection has not been used to obtain Nullstellensatz degree lower bounds, however, due to the difficulty of proving span program lower bounds.

Lower bounds on Nullstellensatz degree have been proven for sets of polynomials encoding combinatorial principles such as the pigeonhole principle [6], induction principle [20], house-sitting principle [26, 18], matching [19], and pebbling [17]. It seems fair to say that research in algebraic proof complexity soon moved on to stronger systems such as *polynomial calculus* [26, 1], where the proof that 1 lies in the ideal generated by $\mathcal{P} \cup \left\{x_j^2 - x_j \,\middle|\, j \in [n]\right\}$ can be

constructed dynamically by a step-by-step derivation. However, the Nullstellensatz proof system has been the focus of renewed interest in a recent line of works [54, 50, 51, 29] showing that Nullstellensatz lower bounds can be lifted to stronger lower bounds for more powerful computational models using composition with gadgets. The size complexity measure for Nullstellensatz has also received attention in recent papers such as [14, 5].

In this work, we are interested in understanding the relation between size and degree in Nullstellensatz. In this context it is relevant to compare and contrast Nullstellensatz with polynomial calculus as well as with the well-known *resolution* proof system [15], which operates directly on the clauses of a CNF formula and repeatedly derives resolvent clauses $C \vee D$ from clauses of the form $C \vee x$ and $D \vee \overline{x}$ until contradiction, in the form of the empty clause without any literals, is reached. For resolution, size is measured by counting the number of clauses, and *width*, measured as the number of literals in a largest clause in a refutation, plays an analogous role to degree for Nullstellensatz and polynomial calculus.

By way of background, it is not hard to show that for all three proof systems upper bounds on degree/width imply upper bounds on size, in the sense that if a CNF formula over n variables can be refuted in degree/width d, then such a refutation can be carried out in size $n^{O(d)}$. Furthermore, this upper bound has been proven to be tight up to constant factors in the exponent for resolution and polynomial calculus [4], and it follows from [44] that this also holds for Nullstellensatz. In the other direction, it has been shown for resolution and polynomial calculus that strong enough lower bounds on degree/width imply lower bounds on size [36, 11]. This is known to be false for Nullstellensatz, and the pebbling formulas discussed in more detail later in this paper provide a counter-example [17].

The size lower bounds in terms of degree/width in [36, 11] can be established by transforming refutations in small size to refutations in small degree/width. This procedure blows up the size of the refutations exponentially, however. It is natural to ask whether such a blow-up is necessary or whether it is just an artifact of the proof. More generally, given that a formula has proofs in small size and small degree/width, it is an interesting question whether both measures can be optimized simultaneously, or whether there has to be a trade-off between the two.

For resolution this question was finally answered in [59], which established that there are indeed strong trade-offs between size and width making the size blow-up in [11] unavoidable. For polynomial calculus, the analogous question remains open.

In this paper, we show that there are strong trade-offs between size and degree for Nullstellensatz. We do so by establishing a tight relation between Nullstellensatz refutations of pebbling formulas and reversible pebblings of the graphs underlying such formulas. In order to discuss this connection in more detail, we first need to describe what reversible pebblings are. This brings us to our next topic.

1.2 Pebble Games

In the *pebble game* first studied by Paterson and Hewitt [48], one places pebbles on the vertices of a directed acyclic graph (DAG) G according to the following rules:

- If all (immediate) predecessors of an empty vertex v contain pebbles, a pebble may be placed on v.
- A pebble may be removed from any vertex at any time.

The game starts and ends with the graph being empty, and a pebble should be placed on the (unique) sink of G at some point. The complexity measures to minimize are the total number of pebbles on G at any given time (the *pebbling space*) and the number of moves (the *pebbling time*).

The pebble game has been used to study flowcharts and recursive schemata [48], register allocation [56], time and space as Turing-machine resources [27, 35], and algorithmic time-space trade-offs [25, 57, 55, 58, 60]. In the last two decades, pebble games have seen a revival in the context of proof complexity (see, e.g., [46]), and pebbling has also turned out to be useful for applications in cryptography [30, 2]. An excellent overview of pebbling up to ca. 1980 is given in [49] and some more recent developments are covered in the upcoming survey [47].

Bennett [13] introduced the *reversible pebble game* as part of a broader program [12] aimed at eliminating or reducing energy dissipation during computation. Reversible pebbling has also been of interest in the context of quantum computing. For example, it was noted in [45] that reversible pebble games can be used to capture the problem of "uncomputing" intermediate values in quantum algorithms.

The reversible pebble game adds the requirement that the whole pebbling performed in reverse order should also be a correct pebbling, which means that the rules for pebble placement and removal become symmetric as follows:

- If all predecessors of an empty vertex v contain pebbles, a pebble may be placed on v.
- If all predecessors of a pebbled vertex v contain pebbles, the pebble on v may be removed.

Reversible pebblings have been studied in [43, 39, 38] and have been used to prove time-space trade-offs in reversible simulation of irreversible computation in [42, 40, 61, 16]. In a different context, Potechin [52] implicitly used reversible pebbling to obtain lower bounds in monotone space complexity, with the connection made explicit in later works [24, 31]. The paper [23] (to which this overview is indebted) studied the relative power of standard and reversible pebblings with respect to space, and also established PSPACE-hardness results for estimating the minimum space required to pebble graphs (reversibly or not).

1.3 Our Contributions

In this paper, we obtain an exactly tight correspondence between on the one hand reversible pebblings of DAGs and on the other hand Nullstellensatz refutations of pebbling formulas over these DAGs. We show that for any DAG G it holds that G can be reversibly pebbled in time t and space s if and only if there is a Nullstellensatz refutation of the pebbling formula over G in size $t+1$ and degree s. This correspondence holds regardless of the field in which the Nullstellensatz refutation is operating, and so, in particular, it follows that pebbling formulas have exactly the same complexity for Nullstellensatz regardless of the ambient field.

We then revisit the time-space trade-off literature for the standard pebble game, focusing on the papers [21, 22, 41]. The results in these papers do not immediately transfer to the reversible pebble game, and we are not fully able to match the tightness of the results for standard pebbling, but we nevertheless obtain strong time-space trade-off results for the reversible pebble game.

This allows us to derive Nullstellensatz size-degree trade-offs from reversible pebbling time-space trade-offs as follows. Suppose that we have a DAG G such that:

1. G can be reversibly pebbled in time $t_1 \ll t_2$.
2. G can be reversibly pebbled in space $s_1 \ll s_2$.
3. There is no reversible pebbling of G that simultaneously achieves time t_1 and space s_1.

Then for Nullstellensatz refutations of the pebbling formula Peb_G over G (which will be formally defined shortly) we can deduce that:

1. Nullstellensatz can refute Peb_G in size $t_1 + 1 \ll t_2 + 1$.
2. Nullstellensatz can also refute Peb_G in degree $s_1 \ll s_2$.
3. There is no Nullstellensatz refutation of Peb_G that simultaneously achieves size $t_1 + 1$ and degree s_1.

We prove four such trade-off results, which can be found in Section 4. The following theorem is one example of such a result (specifically, it is a simplified version of Theorem 4).

▶ **Theorem 1.** *There is a family of 3-CNF formulas $\{F_n\}_{n=1}^{\infty}$ of size $\Theta(n)$ such that:*
1. *There is a Nullstellensatz refutation of F_n in degree $s_1 = \mathrm{O}\big(\sqrt[6]{n}\log n\big)$.*
2. *There is a Nullstellensatz refutation of F_n of near-linear size and degree $s_2 = \mathrm{O}\big(\sqrt[3]{n}\log n\big)$.*
3. *Any Nullstellensatz refutation of F_n in degree at most $\sqrt[3]{n}$ must have exponential size.*

It should be noted that this is not the first time proof complexity trade-off results have been obtained from pebble games. Pebbling formulas were used in [9, 10] to obtain size-space trade-offs for resolution, and later in [8] also for polynomial calculus. However, the current reductions between pebbling and Nullstellensatz are much stronger in that they go in both directions and are exact even up to additive constants.

With regard to Nullstellensatz, it was shown in [17] that Nullstellensatz degree is lower-bounded by standard pebbling price. This was strengthened in [29], which used the connection between designs and Nullstellensatz degree discussed above to establish that the degree needed to refute a pebbling formula exactly coincides with the reversible pebbling price of the corresponding DAG (which is always at least the standard pebbling price, but can be much larger). Our reduction significantly improves on [29] by constructing a more direct reduction, inspired by [34], that can simultaneously capture both time and space.

1.4 Outline of This Paper

After having discussed the necessary preliminaries in Section 2, we prove the reductions between Nullstellensatz and reversible pebblings in Section 3. In Section 4, we present the size-degree trade-offs for Nullstellensatz we obtain for different degree regimes. Section 5 contains some concluding remarks with suggestions for future directions of research.

2 Preliminaries

All logarithms in this paper are base 2 unless otherwise specified. For a positive integer n we write $[n]$ to denote the set of integers $\{1, 2, \ldots, n\}$.

A *literal* a over a Boolean variable x is either the variable x itself or its negation $\overline{x}$ (a *positive* or *negative* literal, respectively). A *clause* $C = a_1 \vee \cdots \vee a_k$ is a disjunction of literals. A *k-clause* is a clause that contains at most k literals. A formula F in *conjunctive normal form (CNF)* is a conjunction of clauses $F = C_1 \wedge \cdots \wedge C_m$. A *$k$-CNF formula* is a CNF formula consisting of k-clauses. We think of clauses and CNF formulas as sets, so that the order of elements is irrelevant and there are no repetitions. A truth value assignment ρ to the variables of a CNF formula F is satisfying if every clause in F contains a literal that is true under ρ.

2.1 Nullstellensatz

Let $\mathbb{F}$ be any field and let $\vec{x} = \{x_1, \ldots, x_n\}$ be a set of variables. We identify a set of polynomials $\mathcal{P} = \{p_i(\vec{x}) \mid i \in [m]\}$ in the ring $\mathbb{F}[\vec{x}]$ with the statement that all $p_i(\vec{x})$ have a common $\{0,1\}$-valued root. A *Nullstellensatz refutation* of this claim is a syntactic equality

$$\sum_{i=1}^{m} r_i(\vec{x}) \cdot p_i(\vec{x}) + \sum_{j=1}^{n} s_j(\vec{x}) \cdot (x_j^2 - x_j) = 1 \ , \tag{2}$$

where r_i, s_j are also polynomials in $\mathbb{F}[\vec{x}]$. We sometimes refer to the polynomials $p_i(\vec{x})$ as axioms and $(x_j^2 - x_j)$ as Boolean axioms.

As discussed in the introduction, Nullstellensatz can be used as a proof system for CNF formulas by translating a clause $C = \bigvee_{x \in P} x \vee \bigvee_{y \in N} \overline{y}$ to the polynomial $p(C) = \prod_{x \in P}(1 - x) \cdot \prod_{y \in N} y$ and viewing Nullstellensatz refutations of $\{p(C_i) \mid i \in [m]\}$ as refutations of the CNF formula $F = \bigwedge_{i=1}^{m} C_i$.

The *degree* of a Nullstellensatz refutation (1) is $\max\{\deg(r_i(\vec{x}) \cdot p_i(\vec{x})), \deg(s_j(\vec{x}) \cdot (x_j^2 - x_j))\}$. We define the *size* of a refutation (2) to be the total number of monomials encountered when all products of polynomials are expanded out as linear combinations of monomials. To be more precise, let $mSize(p)$ denote the number of monomials in a polynomial p written as a linear combination of monomials. Then the size of a Nullstellensatz refutation on the form (1) is

$$\sum_{i=1}^{m} mSize\big(r_i(\vec{x})\big) \cdot mSize\big(p_i(\vec{x})\big) + \sum_{j=1}^{n} 2 \cdot mSize\big(s_j(\vec{x})\big) \ . \tag{3}$$

This is consistent with how size is defined for the "dynamic version" of Nullstellensatz known as polynomial calculus [26, 1], and also with the general size definitions for so-called algebraic and semialgebraic proof systems in [4, 14, 5].

We remark that this is not the only possible way of measuring size, however. It can be noted that the definition (3) is quite wasteful in that it forces us to represent the proof in a very inefficient way. Other papers in the semialgebraic proof complexity literature, such as [33, 37, 28], instead define size in terms of the polynomials in isolation, more along the lines of

$$\sum_{i=1}^{m} \big(mSize\big(r_i(\vec{x})\big) + mSize\big(p_i(\vec{x})\big)\big) + \sum_{j=1}^{n} \big(mSize\big(s_j(\vec{x})\big) + 2\big) \ , \tag{4}$$

or as the bit size or "any reasonable size" of the representation of all polynomials $r_i(\vec{x}), p_i(\vec{x})$, and $s_j(\vec{x})$.

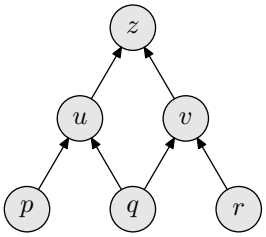
In the end, the difference is not too important since the two measures (3) and (4) are at most a square apart, and for size we typically want to distinguish between polynomial and superpolynomial. In addition, and more importantly, in this paper we will only deal with k-CNF formulas with $k = \mathrm{O}(1)$, and in this setting the two definitions are the same up to a constant factor 2^k. Therefore, we will stick with (3), which matches best how size is measured in the closely related proof systems resolution and polynomial calculus, and which gives the cleanest statements of our results.[1]

When proving lower bounds for algebraic proof systems it is often convenient to consider a *multilinear* setting where refutations are presented in the ring $\mathbb{F}[\vec{x}]/\{x_j^2 - x_j \mid j \in [n]\}$. Since the Boolean axioms $x_j^2 - x_j$ are no longer needed, the refutation (2) can be written simply as

$$\sum_{i=1}^{m} r_i(\vec{x}) \cdot p_i(\vec{x}) = 1 \ , \tag{5}$$

where we assume that all results of multiplications are implicitly multilinearized. It is clear that any refutation on the form (2) remains valid after multilinearization, and so the size and degree measures can only decrease in a multilinear setting. In this paper, we prove our lower bound in our reduction in the multilinear setting and the upper bound in the non-multilinear setting, making the tightly matching results even stronger.

[1] We refer the reader to Section 2.4 in [3] for a more detailed discussion of the definition of proof size in algebraic and semialgebraic proof systems.



(a) Pyramid of height 2.

x_p	$x_p - 1$
$\wedge\ x_q$	$x_q - 1$
$\wedge\ x_r$	$x_r - 1$
$\wedge\ (\overline{x}_p \vee \overline{x}_q \vee x_u)$	$x_p x_q (1 - x_u)$
$\wedge\ (\overline{x}_q \vee \overline{x}_r \vee x_v)$	$x_q x_r (1 - x_v)$
$\wedge\ (\overline{x}_u \vee \overline{x}_v \vee x_z)$	$x_u x_v (1 - x_z)$
$\wedge\ \overline{x}_z$	x_z

(b) Pebbling formula in CNF. **(c)** Polynomial translation.

Figure 1 Example pebbling contradiction for the pyramid graph of height 2.

2.2 Reversible Pebbling and Pebbling Formulas

Throughout this paper $G = (V, E)$ denotes a directed acyclic graph (DAG) of constant fan-in with vertices $V(G) = V$ and edges $E(G) = E$. For an edge $(u, v) \in E$ we say that u is a *predecessor* of v and v a *successor* of u. We write $pred_G(v)$ to denote the sets of all predecessors of v, and drop the subscript when the DAG is clear from context. Vertices with no predecessors/successors are called *sources*/*sinks*. Unless stated otherwise we will assume that all DAGs under consideration have a unique sink z.

A *pebble configuration* on a DAG $G = (V, E)$ is a subset of vertices $\mathbb{P} \subseteq V$. A *reversible pebbling strategy* for a DAG G with sink z, or a *reversible pebbling* of G for short, is a sequence of pebble configurations $\mathcal{P} = (\mathbb{P}_0, \mathbb{P}_1, \ldots, \mathbb{P}_t)$ such that $\mathbb{P}_0 = \mathbb{P}_t = \emptyset$, $z \in \bigcup_{0 \leq t \leq t} \mathbb{P}_t$, and such that each configuration can be obtained from the previous one by one of the following rules:

1. $\mathbb{P}_{i+1} = \mathbb{P}_i \cup \{v\}$ for $v \notin \mathbb{P}_i$ such that $pred_G(v) \subseteq \mathbb{P}_i$ (a *pebble placement* on v).
2. $\mathbb{P}_{i+1} = \mathbb{P}_i \setminus \{v\}$ for $v \in \mathbb{P}_i$ such that $pred_G(v) \subseteq \mathbb{P}_i$ (a *pebble removal* from v).

The *time* of a pebbling $\mathcal{P} = (\mathbb{P}_0, \ldots, \mathbb{P}_t)$ is $\mathsf{time}(\mathcal{P}) = t$ and the *space* is $\mathsf{space}(\mathcal{P}) = \max_{0 \leq t \leq t}\{|\mathbb{P}_t|\}$.

We could also say that a reversible pebbling $\mathcal{P} = (\mathbb{P}_0, \ldots, \mathbb{P}_t)$ should be such that $\mathbb{P}_0 = \emptyset$ and $z \in \mathbb{P}_t$, and define the time of such a pebbling to be $2t$. This is so since once we have reached a configuration containing z we can simply run the pebbling backwards (because of reversibility) until we reach the empty configuration again, and without loss of generality all time- and space-optimal reversible pebblings can be turned into such pebblings. For simplicity, we will often take this viewpoint in what follows.

Pebble games can be encoded in CNF by so-called *pebbling formulas* [11], or *pebbling contradictions*. Given a DAG $G = (V, E)$ with a single sink z, we associate a variable x_v with every vertex v and add clauses encoding that

- the source vertices are all true;
- if all immediate predecessors are true, then truth propagates to the successor;
- but the sink is false.

In short, the pebbling formula over G consists of the clauses $x_v \vee \bigvee_{u \in pred(v)} \neg x_u$ for all $v \in V$ (note that if v is a source $pred(v) = \emptyset$), and the clause $\neg x_z$.

We encode this formula by a set of polynomials in the standard way. Given a set $U \subseteq V$, we denote by x_U the monomial $\prod_{u \in U} x_u$ (in particular, $x_\emptyset = 1$). For every vertex $v \in V$, we have the polynomial

$$A_v := (1 - x_v) \cdot x_{\mathrm{pred}(v)} \ , \tag{6}$$

and for the sink z we also have the polynomial

$$A_{\text{sink}} := x_z \ . \tag{7}$$

See Figure 1 for an illustration, including how the CNF formula is translated to a set of polynomials.

3 Reversible Pebblings and Nullstellensatz Refutations

In this section, we prove the correspondence between the reversible pebbling game on a graph G and Nullstellensatz refutation of the pebbling contradiction of G. Specifically, we prove the following result.

▶ **Theorem 2.** *Let G be a directed acyclic graph with a single sink, let ϕ be the corresponding pebbling contradiction, and let $\mathbb{F}$ be a field. Then, there is a reversible pebbling strategy for G with time at most t and space at most s if and only if there is a Nullstellensatz refutation for ϕ over $\mathbb{F}$ of size at most $t+1$ and degree at most s. Moreover, the same holds for multilinear Nullstellensatz refutations.*

We prove each of the directions of Theorem 2 separately in Sections 3.1 and 3.2 below.

3.1 From Pebbling to Refutation

We start by proving the "only if" direction of Theorem 2. Let

$$\mathbb{P} = (\mathbb{P}_0, \ldots, \mathbb{P}_t) \tag{8}$$

be an optimal reversible pebbling strategy for G. Let $\mathbb{P}_{t'}$ be the first configuration in which there is a pebble on the sink z. Without loss of generality, we may assume that $t = 2 \cdot t'$: if the last $t - t'$ steps were more efficient than the first t' steps, we could have obtained a more efficient strategy by replacing the first t' steps with the (reverse of) the last $t - t'$ steps, and vice versa.

We use $\mathbb{P}$ to construct a Nullstellensatz refutation over $\mathbb{F}$ for the pebbling contradiction ϕ. To this end, we will first construct for each step $i \in [t']$ of $\mathbb{P}$ a Nullstellensatz derivation of the polynomial $x_{\mathbb{P}_{i-1}} - x_{\mathbb{P}_i}$. The sum of all these polynomials is a telescoping sum, and is therefore equal to

$$x_{\mathbb{P}_0} - x_{\mathbb{P}_{t'}} = 1 - x_{\mathbb{P}_{t'}} \ . \tag{9}$$

We will then transform this sum into a Nullstellensatz refutation by adding the polynomial

$$x_{\mathbb{P}_{t'}} = A_{\text{sink}} \cdot x_{\mathbb{P}_{t'} - \{z\}} \ . \tag{10}$$

We turn to constructing the aforementioned derivations. To this end, for every $i \in [t']$, let $v_i \in V$ denote the vertex which was pebbled or unpebbled during the i-th step, i.e., during the transition from $\mathbb{P}_{i-1}$ to $\mathbb{P}_i$. Then, we know that in both configurations $\mathbb{P}_{i-1}$ and $\mathbb{P}_i$ the predecessors of v_i have pebbles on them, i.e., $\text{pred}(v) \subseteq \mathbb{P}_{i-1}, \mathbb{P}_i$. Let us denote by $R_i = \mathbb{P}_i - \{v_i\} - \text{pred}(v_i)$ the set of other vertices that have pebbles during the i-th step. Finally, let p_i be a number that equals to 1 if v_i was pebbled during the i-th step, and equals to -1 if v_i was unpebbled. Now, observe that

$$x_{\mathbb{P}_{i-1}} - x_{\mathbb{P}_i} = p_i \cdot x_{\mathbb{P}_{i-1}}(1 - x_{v_i}) = p_i \cdot x_{R_i} A_{v_i} \ , \tag{11}$$

where the last step follows since the predecessors of v_i are necessarily in $\mathbb{P}_{i-1}$. Therefore, our final refutation for ϕ is

$$\begin{aligned}\sum_{i=1}^{t'} A_{v_i} \cdot p_i \cdot x_{R_i} + A_{\text{sink}} \cdot x_{\mathbb{P}_{t'}-\{z\}} &= x_{\mathbb{P}_{t'}} + \sum_{i=1}^{t'} x_{\mathbb{P}_{i-1}} - x_{\mathbb{P}_i} \\ &= x_{\mathbb{P}_{t'}} + (x_{\mathbb{P}_0} - x_{\mathbb{P}_{t'}}) \\ &= x_{\mathbb{P}_{t'}} + (1 - x_{\mathbb{P}_{t'}}) = 1 \ .\end{aligned} \tag{12}$$

Note, in fact, it is a multilinear Nullstellensatz refutation, since it contains only multilinear monomials and does not use the Boolean axioms. It remains to analyze its degree and size.

For the degree, observe that every monomial in the proof is of the form $x_{\mathbb{P}_i}$, and the degree of each such monomial is exactly the number of pebbles used in the corresponding configuration. It follows that the maximal degree is exactly the space of the pebbling strategy $\mathbb{P}$.

Let us turn to considering the size. Observe that for each of the configurations $\mathbb{P}_1, \ldots, \mathbb{P}_{t'}$, the refutation contains exactly two monomials: for all $i \in [t'-1]$, one monomial for $\mathbb{P}_i$ is generated in the i-th step, and another in the $(i+1)$-th step, and for the configuration $\mathbb{P}_{t'}$ the second monomial is generated when we add $A_{\text{sink}} \cdot x_{\mathbb{P}_{t'}-\{z\}}$. In addition, the refutation contains exactly one monomial for the configuration $\mathbb{P}_0$, which is generated in the first step. Hence, the total number of monomials generated in the refutation is exactly $2 \cdot t' + 1 = t + 1$, as required.

3.2 From Refutation to Pebbling

We turn to prove the "if" direction of Theorem 2. We note that it suffices to prove it for multilinear Nullstellensatz refutations, since every standard Nullstellensatz refutation implies the existence of a multilinear one with at most the same size and degree. Let

$$\sum_{v \in V} A_v \cdot Q_v + A_{\text{sink}} \cdot Q_{\text{sink}} = 1 \tag{13}$$

be a multilinear Nullstellensatz refutation of ϕ over $\mathbb{F}$ of degree s. We use this refutation to construct a reversible pebbling strategy $\mathbb{P}$ for G.

To this end, we construct a "configuration graph" $\mathbb{C}$, whose vertices consist of all possible configurations of at most s pebbles on G (i.e., the vertices will be all subsets of V of size at most s). The edges of $\mathbb{C}$ will be determined by the polynomials Q_v of the refutation, and every edge $\{U_1, U_2\}$ in $\mathbb{C}$ will constitute a legal move in the reversible pebbling game (i.e., it will be legal to move from U_1 to U_2 and vice versa). We will show that $\mathbb{C}$ contains a path from the empty configuration $\emptyset$ to a configuration U_z that contains the sink z, and our pebbling strategy will be generated by walking on this path from $\emptyset$ to U_z and back.

The edges of the configuration graph $\mathbb{C}$ are defined as follows: Let $v \in V$ be a vertex of G, and let q be a monomial of Q_v that *does not contain* x_v. Let $W \subseteq V$ be the set of vertices such that $q = x_W$ (such a set W exists since the refutation is multilinear). Then, we put an edge e_q in $\mathbb{C}$ that connects $W \cup \text{pred}(v)$ and $W \cup \text{pred}(v) \cup \{v\}$ (we allow parallel edges). It is easy to see that the edge e_q connects configurations of size at most s, and that it indeed constitutes a legal move in the reversible pebbling game. We note that $\mathbb{C}$ is a bipartite graph: to see it, note that every edge e_q connects a configuration of an odd size to a configuration of an even size.

For the sake of the analysis, we assign the edge e_q a weight in $\mathbb{F}$ that is equal to coefficient of q in Q_v. We define *the weight of a configuration* U to be the sum of the weights of all the edges that touch U (where the addition is done in the field $\mathbb{F}$). We use the following technical claim, which we prove at the end of this section.

▷ Claim 3. Let $U \subseteq V$ be a configuration in $\mathbb{C}$ that does not contain the sink z. If U is empty, then its weight is 1. Otherwise, its weight is 0.

We now show how to construct the required pebbling strategy $\mathbb{P}$ for G. To this end, we first prove that there is a path in $\mathbb{C}$ from the empty configuration to a configuration that contains the sink z. Suppose for the sake of contradiction that this is not the case, and let $\mathbb{H}$ be the connected component of $\mathbb{C}$ that contains the empty configuration. Our assumption says that none of the configurations in $\mathbb{H}$ contains z.

The connected component $\mathbb{H}$ is bipartite since $\mathbb{C}$ is bipartite. Without loss of generality, assume that the empty configuration is in the left-hand side of $\mathbb{H}$. Clearly, the sum of the weights of the configurations on the left-hand side should be equal to the corresponding sum on the right-hand side, since they are both equal to the sum of the weights of the edges in $\mathbb{H}$. However, the sum of the weights of the configurations on the right-hand side is 0 (since all these weights are 0 by Claim 3), while the sum of the weights of the left-hand side is 1 (again, by Claim 3). We reached a contradiction, and therefore $\mathbb{H}$ must contain some configuration U_z that contains the sink z.

Next, let $\emptyset = \mathbb{P}_0, \mathbb{P}_1, \ldots, \mathbb{P}_{t'} = U_z$ be a path from the empty configuration to U_z. Our reversible pebbling strategy for G is

$$\mathbb{P} = (\mathbb{P}_0, \ldots, \mathbb{P}_{t'-1}, \mathbb{P}_{t'}, \mathbb{P}_{t'-1}, \ldots, \mathbb{P}_0) \quad . \tag{14}$$

This is a legal pebbling strategy since, as noted above, every edge of $\mathbb{C}$ constitutes a legal move of the reversible pebbling game. The strategy $\mathbb{P}$ uses space s, since all the configurations in $\mathbb{C}$ contain at most s pebbles by definition. The time of $\mathbb{P}$ is $t = 2 \cdot t'$. It therefore remains to show that the size of the Nullstellensatz refutation from Equation 13 is at least $t + 1$.

To this end, note that every edge e_q in the path corresponds to some monomial q in some polynomial Q_v. When the monomial q is multiplied by the axiom A_v, it generates two monomials in the proof: the monomial $q \cdot x_{\mathrm{pred}(v)}$ and the monomial $q \cdot x_{\mathrm{pred}(v)} \cdot x_v$. Hence, the Nullstellensatz refutation contains at least $2 \cdot t'$ monomials that correspond to edges from the path. In addition, the product $A_{\mathrm{sink}} \cdot Q_{\mathrm{sink}}$ must contains at least one monomial, since the refutation must use the sink axiom A_{sink} (because ϕ without this axiom is not a contradiction). It follows that the refutation contains at least $2 \cdot t' + 1 = t + 1$ monomials, as required. We conclude this section by proving Claim 3.

Proof of Claim 3. We start by introducing some terminology. First, observe that a monomial m may be generated multiple times in the refutation of Equation 13, and we refer to each time it is generated as an *occurrence* of m. We say that an occurrence of m is *generated by a monomial q_v of Q_v* if it is generated by the product $A_v \cdot q_v$. Throughout the proof, we identify a configuration U with the monomial x_U.

We first prove the claim for the non-empty case. Let $U \subseteq V$ be a non-empty configuration. We would like to prove the weight of U is 0. Recall that the weight of U is the sum of the coefficients of the occurrences of U that are generated by monomials q_v *that do not contain the corresponding vertex* v. Observe that Equation 13 implies that the sum of the coefficients of *all* the occurrences of U is 0: the coefficient of U on the right-hand side is 0, and it must be equal to the coefficient of U on the left-hand side, which is the sum of the coefficients of all the occurrences.

To complete the proof, we argue that every monomial q_v that does contain the vertex v contributes 0 to that sum. Let q_v be a monomial of Q_v that contains the vertex v and

generates an occurrence of U. Let α be the coefficient of q. Then, it must hold that

$$\begin{aligned} A_v \cdot q_v &= x_{\mathrm{pred}(v)} \cdot q_v - x_v \cdot x_{\mathrm{pred}(v)} \cdot q_v \\ &= x_{\mathrm{pred}(v)} \cdot q_v - x_{\mathrm{pred}(v)} \cdot q_v \\ &= \alpha \cdot x_U - \alpha \cdot x_U \ , \end{aligned} \tag{15}$$

where the second equality holds since we q_v contains v and we are working with a multilinear refutation, and the third equality holds since we assumed that q_v generates an occurrence of U. It follows that q_v generates two occurrences of U, one with coefficient α and one with coefficient $-\alpha$, and therefore it contributes 0 to the sum of the coefficients of all the occurrences of U.

We have shown that the sum of the coefficients of all the occurrences of U is 0, and that the occurrences generated by monomials q_v that contain v contribute 0 to this sum, and therefore the sum of coefficients of occurrences that are generated by monomials q_v that do not contain v must be 0, as required. In the case that U is the empty configuration, the proof is identical, except that the sum of the coefficients of all occurrences is 1, since the coefficient of $\emptyset$ is 1 on the right hand side of Equation 13. ◁

4 Nullstellensatz Trade-offs from Reversible Pebbling

In this section we present the Nullstellensatz size-degree trade-offs we obtain for different degree regimes. Let us first recall what is known with regards to degree and size. In what follows, a Nullstellensatz refutation of a CNF formula F refers to a Nullstellensatz refutation of the translation of F to polynomials. As mentioned in the introduction, if a CNF formula over n variables can be refuted in degree d then it can be refuted in simultaneous degree d and size $n^{\mathrm{O}(d)}$. However, for Nullstellensatz it is not the case that strong enough degree lower bounds imply size lower bounds.

A natural question is whether for any given function $d_1(n)$ there is a family of CNF formulas $\{F_n\}_{n=1}^{\infty}$ of size $\Theta(n)$ such that

1. F_n has a Nullstellensatz refutation $d_1(n)$;
2. F_n has a Nullstellensatz refutation of (close to) linear size and degree $d_2(n) \gg d_1(n)$;
3. Any Nullstellensatz refutation of F_n in degree only slightly below $d_2(n)$ must have size nearly $n^{d_1(n)}$.

We present explicit constructions of formulas providing such trade-offs in several different parameter regimes. We first show that there are formulas that require exponential size in Nullstellensatz if the degree is bounded by some polynomial function, but if we allow slightly larger degree there is a nearly linear size proof.

▶ **Theorem 4.** *There is a family of explicitly constructible unsatisfiable 3-CNF formulas $\{F_n\}_{n=1}^{\infty}$ of size $\Theta(n)$ such that:*

1. *There is a Nullstellensatz refutation of F_n in degree $d_1 = \mathrm{O}\big(\sqrt[6]{n} \log n\big)$.*
2. *For any constant $\epsilon > 0$, there is a Nullstellensatz refutation of F_n of size $\mathrm{O}(n^{1+\epsilon})$ and degree $d_2 = \mathrm{O}\big(d_1 \cdot \sqrt[6]{n}\big) = \mathrm{O}\big(\sqrt[3]{n} \log n\big)$.*
3. *There exists a constant $K > 0$ such that any Nullstellensatz refutation of F_n in degree at most $d = Kd_2/\log n = \mathrm{O}\big(\sqrt[3]{n}\big)$ must have size $\big(\sqrt[6]{n}\big)!$.*

We also analyse a family of formulas that can be refuted in close to logarithmic degree and show that even if we allow up to a certain polynomial degree, the Nullstellensatz size required is superpolynomial.

▶ **Theorem 5.** *Let $\delta > 0$ be an arbitrarily small positive constant and let $g(n)$ be any arbitrarily slowly growing monotone function $\omega(1) = g(n) \leq n^{1/4}$. Then there is a family of explicitly constructible unsatisfiable 3-CNF formulas $\{F_n\}_{n=1}^{\infty}$ of size $\Theta(n)$ such that:*

1. *There is a Nullstellensatz refutation of F_n in degree $d_1 = g(n)\log(n)$.*
2. *For any constant $\epsilon > 0$, there is a Nullstellensatz refutation of F_n of size $\mathrm{O}(n^{1+\epsilon})$ and degree*

$$d_2 = \mathrm{O}\big(d_1 \cdot n^{1/2}/g(n)^2\big) = \mathrm{O}\big(n^{1/2}\log n/g(n)\big).$$

3. *Any Nullstellensatz refutation of F_n in degree at most*

$$d = \mathrm{O}\big(d_2/n^{\delta}\log n\big) = \mathrm{O}\big(n^{1/2-\delta}/g(n)\big)$$

must have size superpolynomial in n.

Still in the small-degree regime, we present a very robust trade-off in the sense that superpolynomial size lower bound holds for degree from $\log^2(n)$ to $n/\log(n)$.

▶ **Theorem 6.** *There is a family of explicitly constructible unsatisfiable 3-CNF formulas $\{F_n\}_{n=1}^{\infty}$ of size $\Theta(n)$ such that:*

1. *There is a Nullstellensatz refutation of F_n in degree $d_1 = \mathrm{O}(\log^2 n)$.*
2. *For any constant $\delta > 0$, there is a Nullstellensatz refutation of F_n of size $\mathrm{O}(n)$ and degree*

$$d_2 = \mathrm{O}(d_1 \cdot n/\log^{3-\delta} n) = \mathrm{O}(n/\log^{1-\delta} n).$$

3. *There exists a constant $K > 0$ such that any Nullstellensatz refutation of F_n in degree at most $d = Kd_2/\log^{\delta} n = O(n/\log n)$ must have size $n^{\Omega(\log\log n)}$.*

Finally, we study a family of formulas that have Nullstellensatz refutation of quadratic size and that present a smooth size-degree trade-off.

▶ **Theorem 7.** *There is a family of explicitly constructible unsatisfiable 3-CNF formulas $\{F_n\}_{n=1}^{\infty}$ of size $\Theta(n)$ such that any Nullstellensatz refutation of F_n that optimizes size given degree constraint $d = n^{\Theta(1)}$ (and less than n) has size $\Theta\big(n^2/d\big)$.*

We prove these results by obtaining the analogous time-space trade-offs for reversible pebbling and then applying the tight correspondence between size and degree in Nullstellensatz and time and space in reversible pebbling. We differ the reader to the upcoming full version for the details.

5 Concluding Remarks

In this paper we prove that size and degree of Nullstellensatz refutations in any field of pebbling formulas are exactly captured by time and space of the reversible pebble game on the underlying graph. This allows us to prove a number of strong size-degree trade-offs for Nullstellensatz. To the best of our understanding no such results have been known previously.

The most obvious, and also most interesting, open question is whether there are also size-degree trade-offs for the stronger polynomial calculus proof system. Such trade-offs cannot be exhibited by the pebbling formulas considered in this work, since such formulas have small-size low-degree polynomial calculus refutations, but the formulas exhibiting size-width trade-offs for resolution [59] appear to be natural candidates.

Another interesting question is whether the tight relation between Nullstellensatz and reversible pebbling could make it possible to prove even sharper trade-offs for size versus degree in Nullstellensatz, where just a small constant drop in the degree would lead to an

exponential blow-up in size. Such results for pebbling time versus space are known for the standard pebble game, e.g., in [32]. It is conceivable that a similar idea could be applied to the reversible pebbling reductions in [23], but it is not obvious whether just adding a small amount of space makes it possible to carry out the reversible pebbling time-efficiently enough.

Finally, it can be noted that our results crucially depend on that we are in a setting with variables only for positive literals. For polynomial calculus it is quite common to consider the stronger setting with "twin variables" for negated literals (as in the generalization of polynomial calculus in [26] to *polynomial calculus resolution* in [1]). It would be nice to generalize our size-degree trade-offs for Nullstellensatz to this setting, but it is not obvious whether the reductions in the current work could be made to work or not.

References

1 Michael Alekhnovich, Eli Ben-Sasson, Alexander A. Razborov, and Avi Wigderson. Space Complexity in Propositional Calculus. *SIAM Journal on Computing*, 31(4):1184–1211, 2002. Preliminary version in *STOC '00*.

2 Joël Alwen and Vladimir Serbinenko. High Parallel Complexity Graphs and Memory-Hard Functions. In *Proceedings of the 47th Annual ACM Symposium on Theory of Computing (STOC '15)*, pages 595–603, June 2015.

3 Albert Atserias and Tuomas Hakoniemi. Size-Degree Trade-Offs for Sums-of-Squares and Positivstellensatz Proofs. Technical report, arXiv.org, November 2018. `arXiv:1811.01351`.

4 Albert Atserias, Massimo Lauria, and Jakob Nordström. Narrow Proofs May Be Maximally Long. *ACM Transactions on Computational Logic*, 17(3):19:1–19:30, May 2016. Preliminary version in *CCC '14*.

5 Albert Atserias and Joanna Ochremiak. Proof Complexity Meets Algebra. *ACM Transactions on Computational Logic*, 20:1:1–1:46, February 2019. Preliminary version in *ICALP '17*.

6 Paul Beame, Stephen A. 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, August 1998. Preliminary version in *STOC '95*.

7 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 Annual IEEE Symposium on Foundations of Computer Science (FOCS '94)*, pages 794–806, November 1994.

8 Chris Beck, Jakob Nordström, and Bangsheng Tang. Some Trade-off Results for Polynomial Calculus. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing (STOC '13)*, pages 813–822, May 2013.

9 Eli Ben-Sasson. Size-Space Tradeoffs for Resolution. *SIAM Journal on Computing*, 38(6):2511–2525, May 2009. Preliminary version in *STOC '02*.

10 Eli Ben-Sasson and Jakob Nordström. Understanding Space in Proof Complexity: Separations and Trade-offs via Substitutions. In *Proceedings of the 2nd Symposium on Innovations in Computer Science (ICS '11)*, pages 401–416, January 2011.

11 Eli Ben-Sasson and Avi Wigderson. Short Proofs are Narrow—Resolution Made Simple. *Journal of the ACM*, 48(2):149–169, March 2001. Preliminary version in *STOC '99*.

12 Charles H. Bennett. Logical Reversibility of Computation. *IBM Journal of Research and Development*, 17(6):525–532, November 1973.

13 Charles H. Bennett. Time/Space Trade-offs for Reversible Computation. *SIAM Journal on Computing*, 18(4):766–776, August 1989.

14 Christoph Berkholz. The Relation between Polynomial Calculus, Sherali-Adams, and Sum-of-Squares Proofs. In *Proceedings of the 35th Symposium on Theoretical Aspects of Computer Science (STACS '18)*, volume 96 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 11:1–11:14, February 2018.

15 Archie Blake. *Canonical Expressions in Boolean Algebra*. PhD thesis, University of Chicago, 1937.

16 Harry Buhrman, John Tromp, and Paul Vitányi. Time and Space Bounds for Reversible Simulation. *Journal of physics A: Mathematical and general*, 34:6821–6830, 2001. Preliminary version in *ICALP '01*.

17 Joshua Buresh-Oppenheim, Matthew Clegg, Russell Impagliazzo, and Toniann Pitassi. Homogenization and the Polynomial Calculus. *Computational Complexity*, 11(3-4):91–108, 2002. Preliminary version in *ICALP '00*.

18 Samuel R. Buss. Lower Bounds on Nullstellensatz Proofs via Designs. In *Proof Complexity and Feasible Arithmetics*, volume 39 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 59–71. American Mathematical Society, 1998. Available at `http://www.math.ucsd.edu/~sbuss/ResearchWeb/designs/`.

19 Samuel R. Buss, Russell Impagliazzo, Jan Krajíček, Pavel Pudlák, Alexander A. Razborov, and Jiri Sgall. Proof Complexity in Algebraic Systems and Bounded Depth Frege Systems with Modular Counting. *Computational Complexity*, 6(3):256–298, 1997.

20 Samuel R. Buss and Toniann Pitassi. Good Degree Bounds on Nullstellensatz Refutations of the Induction Principle. *Journal of Computer and System Sciences*, 2(57):162–171, October 1998. Preliminary version in *CCC '96*.

21 David A. Carlson and John E. Savage. Graph Pebbling with Many Free Pebbles Can Be Difficult. In *Proceedings of the 12th Annual ACM Symposium on Theory of Computing (STOC '80)*, pages 326–332, 1980.

22 David A. Carlson and John E. Savage. Extreme Time-Space Tradeoffs for Graphs with Small Space Requirements. *Information Processing Letters*, 14(5):223–227, 1982.

23 Siu Man Chan, Massimo Lauria, Jakob Nordström, and Marc Vinyals. Hardness of Approximation in PSPACE and Separation Results for Pebble Games (Extended Abstract). In *Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science (FOCS '15)*, pages 466–485, October 2015.

24 Siu Man Chan and Aaron Potechin. Tight Bounds for Monotone Switching Networks via Fourier Analysis. *Theory of Computing*, 10:389–419, October 2014. Preliminary version in *STOC '12*.

25 Ashok K. Chandra. Efficient Compilation of Linear Recursive Programs. In *Proceedings of the 14th Annual Symposium on Switching and Automata Theory (SWAT '73)*, pages 16–25, 1973.

26 Matthew Clegg, Jeffery Edmonds, and Russell Impagliazzo. Using the Groebner Basis Algorithm to Find Proofs of Unsatisfiability. In *Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC '96)*, pages 174–183, May 1996.

27 Stephen A. Cook. An Observation on Time-Storage Trade Off. *Journal of Computer and System Sciences*, 9(3):308–316, 1974. Preliminary version in *STOC '73*.

28 Stefan S. Dantchev, Barnaby Martin, and Martin Rhodes. Tight Rank Lower Bounds for the Sherali–Adams Proof System. *Theoretical Computer Science*, 410(21–23):2054–2063, May 2009.

29 Susanna F. de Rezende, Or Meir, Jakob Nordström, Toniann Pitassi, Robert Robere, and Marc Vinyals. Lifting with Simple Gadgets and Applications to Circuit and Proof Complexity. Manuscript in preparation, 2019.

30 Cynthia Dwork, Moni Naor, and Hoeteck Wee. Pebbling and Proofs of Work. In *Proceedings of the 25th Annual International Cryptology Conference (CRYPTO '05)*, volume 3621 of *Lecture Notes in Computer Science*, pages 37–54. Springer, August 2005.

31 Yuval Filmus, Toniann Pitassi, Robert Robere, and Stephen A Cook. Average Case Lower Bounds for Monotone Switching Networks. In *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS '13)*, pages 598–607, November 2013.

32 John R. Gilbert, Thomas Lengauer, and Robert Endre Tarjan. The Pebbling Problem is Complete in Polynomial Space. *SIAM Journal on Computing*, 9(3):513–524, August 1980. Preliminary version in *STOC '79*.

33 Dima Grigoriev, Edward A. Hirsch, and Dmitrii V. Pasechnik. Exponential Lower Bound for Static Semi-algebraic Proofs. In *Proceedings of the 29th International Colloquium on Automata, Languages and Programming (ICALP '02)*, volume 2380 of *Lecture Notes in Computer Science*, pages 257–268. Springer, July 2002.

34 Mika Göös, Pritish Kamath, Robert Robere, and Dmitry Sokolov. Adventures in Monotone Complexity and TFNP. Technical Report TR18-163, Electronic Colloquium on Computational Complexity (ECCC), September 2018.

35 John Hopcroft, Wolfgang Paul, and Leslie Valiant. On Time Versus Space. *Journal of the ACM*, 24(2):332–337, April 1977. Preliminary version in *FOCS '75*.

36 Russell Impagliazzo, Pavel Pudlák, and Jiří Sgall. Lower Bounds for the Polynomial Calculus and the Gröbner Basis Algorithm. *Computational Complexity*, 8(2):127–144, 1999.

37 Arist Kojevnikov and Dmitry Itsykson. Lower Bounds of Static Lovász–Schrijver Calculus Proofs for Tseitin Tautologies. In *Proceedings of the 33rd International Colloquium on Automata, Languages and Programming (ICALP '06)*, volume 4051 of *Lecture Notes in Computer Science*, pages 323–334. Springer, July 2006.

38 Balagopal Komarath, Jayalal Sarma, and Saurabh Sawlani. Pebbling meets coloring: Reversible pebble game on trees. *Journal of Computer and System Sciences*, 91:33–41, 2018. `doi:10.1016/j.jcss.2017.07.009`.

39 Richard Královič. Time and Space Complexity of Reversible Pebbling. *RAIRO – Theoretical Informatics and Applications*, 38(02):137–161, April 2004.

40 Klaus-Jörn Lange, Pierre McKenzie, and Alain Tapp. Reversible Space Equals Deterministic Space. *Journal of Computer and System Sciences*, 60(2):354–367, April 2000.

41 Thomas Lengauer and Robert Endre Tarjan. Asymptotically Tight Bounds on Time-Space Trade-offs in a Pebble Game. *Journal of the ACM*, 29(4):1087–1130, October 1982. Preliminary version in *STOC '79*.

42 Ming Li, John Tromp, and Paul Vitányi. Reversible Simulation of Irreversible Computation. *Physica D: Nonlinear Phenomena*, 120(1–2):168–176, September 1998.

43 Ming Li and Paul Vitányi. Reversibility and Adiabatic Computation: Trading Time and Space for Energy. *Proceedings of the Royal Society of London, Series A*, 452(1947):769–789, April 1996.

44 Jesús A. De Loera, Jon Lee, Susan Margulies, and Shmuel Onn. Expressing Combinatorial Problems by Systems of Polynomial Equations and Hilbert's Nullstellensatz. *Combinatorics, Probability and Computing*, 18(4):551–582, July 2009.

45 Giulia Meuli, Mathias Soeken, Martin Roetteler, Nikolaj Bjørner, and Giovanni De Micheli. Reversible Pebbling Game for Quantum Memory Management. *CoRR*, abs/1904.02121, 2019. `arXiv:1904.02121`.

46 Jakob Nordström. Pebble Games, Proof Complexity and Time-Space Trade-offs. *Logical Methods in Computer Science*, 9(3):15:1–15:63, September 2013.

47 Jakob Nordström. New Wine into Old Wineskins: A Survey of Some Pebbling Classics with Supplemental Results. Manuscript in preparation. To appear in *Foundations and Trends in Theoretical Computer Science*. Current draft version available at `http://www.csc.kth.se/~jakobn/research/`, 2019.

48 Michael S. Paterson and Carl E. Hewitt. Comparative Schematology. In *Record of the Project MAC Conference on Concurrent Systems and Parallel Computation*, pages 119–127, 1970.

49 Nicholas Pippenger. Pebbling. Technical Report RC8258, IBM Watson Research Center, 1980. in Proceedings of the 5th IBM Symposium on Mathematical Foundations of Computer Science, Japan.

50 Toniann Pitassi and Robert Robere. Strongly Exponential Lower Bounds for Monotone Computation. In *Proceedings of the 49th Annual ACM Symposium on Theory of Computing (STOC '17)*, pages 1246–1255, June 2017.

51 Toniann Pitassi and Robert Robere. Lifting Nullstellensatz to Monotone Span Programs over Any Field. In *Proceedings of the 50th Annual ACM Symposium on Theory of Computing (STOC '18)*, pages 1207–1219, June 2018.

52 Aaron Potechin. Bounds on Monotone Switching Networks for Directed Connectivity. In *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science (FOCS '10)*, pages 553–562, October 2010.

53 Pavel Pudlák and Jirí Sgall. Algebraic Models of Computation and Interpolation for Algebraic Proof Systems. In *Proof Complexity and Feasible Arithmetics*, volume 39 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 279–296. American Mathematical Society, 1998. Available at `http://users.math.cas.cz/~pudlak/span.pdf`.

54 Robert Robere, Toniann Pitassi, Benjamin Rossman, and Stephen A. Cook. Exponential Lower Bounds for Monotone Span Programs. In *Proceedings of the 57th Annual IEEE Symposium on Foundations of Computer Science (FOCS '16)*, pages 406–415, October 2016.

55 John E. Savage and Sowmitri Swamy. Space-Time Tradeoffs for Oblivious Interger Multiplications. In *Proceedings of the 6th International Colloquium on Automata, Languages and Programming (ICALP '79)*, pages 498–504, 1979.

56 Ravi Sethi. Complete Register Allocation Problems. *SIAM Journal on Computing*, 4(3):226–248, September 1975.

57 Sowmitri Swamy and John E. Savage. Space-Time Trade-offs on the FFT-algorithm. Technical Report CS-31, Brown University, 1977.

58 Sowmitri Swamy and John E. Savage. Space-Time Tradeoffs for Linear Recursion. *Mathematical Systems Theory*, 16(1):9–27, 1983.

59 Neil Thapen. A Trade-off Between Length and Width in Resolution. *Theory of Computing*, 12(5):1–14, August 2016.

60 Martin Tompa. Time-Space Tradeoffs for Computing Functions, Using Connectivity Properties of Their Circuits. In *Proceedings of the 10th annual ACM symposium on Theory of computing (STOC '78)*, pages 196–204, 1978.

61 Ryan Williams. Space-Efficient Reversible Simulations. Technical report, Cornell University, 2000. Available at `http://web.stanford.edu/~rrwill/spacesim9_22.pdf`.

Stronger Connections Between Circuit Analysis and Circuit Lower Bounds, via PCPs of Proximity

Lijie Chen
MIT, Cambridge, MA, USA

R. Ryan Williams
MIT, Cambridge, MA, USA

Abstract

We considerably sharpen the known connections between circuit-analysis algorithms and circuit lower bounds, show intriguing equivalences between the analysis of weak circuits and (apparently difficult) circuits, and provide strong new lower bounds for approximately computing Boolean functions with depth-two neural networks and related models.

- We develop approaches to proving $\mathsf{THR} \circ \mathsf{THR}$ lower bounds (a notorious open problem), by connecting algorithmic analysis of $\mathsf{THR} \circ \mathsf{THR}$ to the *provably weaker* circuit classes $\mathsf{THR} \circ \mathsf{MAJ}$ and $\mathsf{MAJ} \circ \mathsf{MAJ}$, where *exponential* lower bounds have long been known. More precisely, we show equivalences between algorithmic analysis of $\mathsf{THR} \circ \mathsf{THR}$ and these weaker classes. The ε-error CAPP problem asks to approximate the acceptance probability of a given circuit to within additive error ε; it is the "canonical" derandomization problem. We show:
  - There is a non-trivial ($2^n/n^{\omega(1)}$ time) $1/\operatorname{poly}(n)$-error CAPP algorithm for $\operatorname{poly}(n)$-size $\mathsf{THR} \circ \mathsf{THR}$ circuits if and only if there is such an algorithm for $\operatorname{poly}(n)$-size $\mathsf{MAJ} \circ \mathsf{MAJ}$.
  - There is a $\delta > 0$ and a non-trivial SAT (δ-error CAPP) algorithm for $\operatorname{poly}(n)$-size $\mathsf{THR} \circ \mathsf{THR}$ circuits *if and only* if there is such an algorithm for $\operatorname{poly}(n)$-size $\mathsf{THR} \circ \mathsf{MAJ}$.

  Similar results hold for depth-d linear threshold circuits and depth-d MAJORITY circuits.

  These equivalences are proved via new simulations of THR circuits by circuits with MAJ gates.
- We strengthen the connection between non-trivial derandomization (non-trivial CAPP algorithms) for a circuit class $\mathcal{C}$, and circuit lower bounds against $\mathcal{C}$. Previously, [Ben-Sasson and Viola, ICALP 2014] (following [Williams, STOC 2010]) showed that for any polynomial-size class $\mathcal{C}$ closed under projections, non-trivial ($2^n/n^{\omega(1)}$ time) CAPP for $\mathsf{OR}_{\operatorname{poly}(n)} \circ \mathsf{AND}_3 \circ \mathcal{C}$ yields $\mathsf{NEXP} \not\subset \mathcal{C}$. We apply *Probabilistic Checkable Proofs of Proximity* in a new way to show it would suffice to have a non-trivial CAPP algorithm for either $\oplus_2 \circ \mathcal{C}$, $\mathsf{AND}_2 \circ \mathcal{C}$ or $\mathsf{OR}_2 \circ \mathcal{C}$.
- A direct corollary of the first two bullets is that $\mathsf{NEXP} \not\subset \mathsf{THR} \circ \mathsf{THR}$ would follow from either:
  - a non-trivial δ-error CAPP (or SAT) algorithm for $\operatorname{poly}(n)$-size $\mathsf{THR} \circ \mathsf{MAJ}$ circuits, or
  - a non-trivial $1/\operatorname{poly}(n)$-error CAPP algorithm for $\operatorname{poly}(n)$-size $\mathsf{MAJ} \circ \mathsf{MAJ}$ circuits.
- Applying the above machinery, we extend lower bounds for depth-two neural networks and related models [R. Williams, CCC 2018] to weak *approximate* computations of Boolean functions. For example, for arbitrarily small $\varepsilon > 0$, we prove there are Boolean functions f computable in nondeterministic $n^{\log n}$ time such that (for infinitely many n) every polynomial-size depth-two neural network N on n inputs (with sign or ReLU activation) must satisfy $\max_{x \in \{0,1\}^n} |N(x) - f(x)| > 1/2 - \varepsilon$. That is, short linear combinations of ReLU gates fail miserably at computing f to within close precision. Similar results are proved for linear combinations of $\mathsf{ACC} \circ \mathsf{THR}$ circuits, and linear combinations of low-degree $\mathbb{F}_p$ polynomials. These results constitute further progress towards $\mathsf{THR} \circ \mathsf{THR}$ lower bounds.

2012 ACM Subject Classification Theory of computation → Circuit complexity

Keywords and phrases PCP of Proximity, Circuit Lower Bounds, Derandomization, Threshold Circuits, ReLU

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.19

Funding Supported by NSF CCF-1741615 (Common Links in Algorithms and Complexity) and a Google Faculty Research Award.

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka; Article No. 19; pp. 19:1–19:43

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

Acknowledgements Part of this work was completed while the authors were visiting the Simons Institute at UC Berkeley, as part of the program on Lower Bounds in Computational Complexity. We thank them for their hospitality and excellent environment. We also thank Josh Alman for helpful last-minute proofreading, and the CCC reviewers for useful comments.

1 Introduction

Recall TC^0 is the class of decision problems that are computable with circuit families of constant depth, composed of MAJORITY and NOT gates. As this class remains the same when "MAJORITY" is replaced by other more expressive functions such as *linear threshold functions* [21, 32], TC^0 naturally captures many mathematical models of neural computing, and contains many natural arithmetic functions (for example, see [9, 38, 26]).

What interesting functions do not have polynomial-size TC^0 circuits? Despite substantial research effort [23, 2, 4, 17, 19, 21, 22, 24, 25, 29, 28, 35, 37, 40, 51, 44, 3, 31] it is consistent with current knowledge that the huge class *nondeterministic exponential time* (NEXP) has polynomial-size $\mathsf{THR} \circ \mathsf{THR}$ circuits, which are depth *two* and can compute arbitrary linear threshold functions at both layers.[1] It seems obvious that "shallow" nets cannot be so powerful, but concrete proofs of their limitations have been elusive.

In 2011, R. Williams [49, 52] proved that NEXP does not have polynomial-size ACC^0 circuits (a presumably weaker circuit class), by showing how circuit lower bounds follow from non-trivial algorithms[2] for problems such as *circuit satisfiability* or *circuit derandomization*. The canonical circuit derandomization problem is CAPP, where the task is to approximate the acceptance probability of a given circuit within an additive constant error (less than $1/3$, say). Along these lines, subsequent works have followed Williams' program [50, 51, 12, 30, 3, 47, 44, 48], and more circuit lower bounds have been proved by either introducing new SAT algorithms, or tightening the algorithms-to-lower-bounds connection. For an example of the latter, Murray and Williams [34] recently showed that nondeterministic *quasi-polynomial time* does not have polynomial-size $\mathsf{ACC}^0 \circ \mathsf{THR}$ circuits using a strengthened connection.

A potential next step in this program would be to prove that NEXP does not have polynomial-size depth-two threshold circuits ($\mathsf{THR} \circ \mathsf{THR}$). Partial progress has already been made: for example, in [44, 3], it is shown that E^{NP} does not have $n^{2-o(1)}$ size $\mathsf{THR} \circ \mathsf{THR}$ circuits. Until now, essentially all lower bounds proven by this program have applied very strong circuit-analysis algorithms, such as circuit satisfiability or #SAT algorithms. It looks difficult to find such strong circuit-analysis algorithms for $\mathsf{THR} \circ \mathsf{THR}$ circuits. Indeed, even for the simpler MAX-k-SAT problem (equivalent to the SAT problem for $\mathsf{MAJ} \circ \mathsf{AND}_k$ circuits), no non-trivial algorithms are known for $k(n) = \omega(\log n)$ (see [8]). For slightly larger circuit classes such as NC^1 (a.k.a. poly(n)-size formulas), it has been conjectured that there are no non-trivial SAT algorithms [1].

An approach based on *derandomizing* a circuit class (finding non-trivial algorithms for CAPP) looks more plausible than one based on SAT solving, because most researchers believe "full" derandomization is possible and that CAPP is in P even for arbitrary circuits. However, there is still a substantial obstacle for proving $\mathsf{NEXP} \not\subset \mathsf{THR} \circ \mathsf{THR}$ via derandomization approaches. While previous works ([30, 13]) have shown that $\mathsf{NEXP} \not\subset \mathcal{C}$ would follow from a non-trivial satisfiability algorithm for $\mathsf{AND}_3 \circ \mathcal{C}$ (i.e., an AND of three $\mathcal{C}$-circuits), the best

[1] See Section 2.1 for formal definitions.

[2] Throughout the paper, we use the term "non-trivial algorithm" to mean that, for every constant $c \geq 1$, the algorithm runs in $2^n/n^{\omega(1)}$ time on circuits of n inputs and n^c gates.

known connection theorem (namely, that of Ben-Sasson and Viola [13]) is that non-trivial derandomization of $\mathsf{OR}_{\mathrm{poly}(n)} \circ \mathsf{AND}_3 \circ \mathcal{C}$ (i.e., a non-trivial CAPP algorithm for a 3-DNF on $\mathrm{poly}(n)$ many $\mathcal{C}$ circuits) implies $\mathsf{NEXP} \not\subset \mathcal{C}$. Finding a tighter correspondence (with no "DNF overheads" in the connection) has been an intriguing open problem.

In this work, applying *Probabilistically Checkable Proofs of Proximity* (PCPPs), we substantially tighten the connection given by Ben-Sasson and Viola, showing that non-trivial derandomization for depth-d TC circuits would directly imply NEXP does not have depth-d TC circuits. That is, in order to show $\mathsf{NEXP} \not\subset \mathsf{THR} \circ \mathsf{THR}$, it suffices to find a non-trivial derandomization of $\mathsf{THR} \circ \mathsf{THR}$. Furthermore, we show that non-trivial derandomization algorithms for $\mathsf{THR} \circ \mathsf{THR}$ is in fact *equivalent* to derandomization for the weaker class $\mathsf{THR} \circ \mathsf{MAJ}$ (tightly) and derandomization for the even weaker class $\mathsf{MAJ} \circ \mathsf{MAJ}$ (with inverse polynomial error). ($\mathsf{THR} \circ \mathsf{MAJ}$ circuits are the special case of $\mathsf{THR} \circ \mathsf{THR}$ circuits where all gates on the bottom layer only compute linear threshold functions with *polynomial* integer weights; $\mathsf{MAJ} \circ \mathsf{MAJ}$ circuits have that restriction on both layers. See Section 2.1 for formal definitions.) Therefore, for our desired lower bounds against $\mathsf{THR} \circ \mathsf{THR}$, it suffices to obtain non-trivial CAPP algorithms for $\mathsf{THR} \circ \mathsf{MAJ}$ or $\mathsf{MAJ} \circ \mathsf{MAJ}$ circuits, for which exponential-size circuit lower bounds have long been known [19, 23].

As an additional application, we apply our new PCPP approach to strengthen recent depth-two neural network lower bounds of R. Williams [48] for approximate computation of Boolean functions. For example, we show that for every $\varepsilon > 0$ and every (non-uniform) polynomial-size family of depth-two neural nets $\{N_n\}$ with sign or ReLU activation functions, there are Boolean functions f in nondeterministic $n^{O(\log^* n)}$ time such that $\max_{x \in \{0,1\}^n} |N_n(x) - f(x)| > 1/2 - \varepsilon$ for infinitely many n. That is, arbitrary linear combinations of ReLU gates fail miserably at computing f to within any close precision. Versions of the PCP theorem are crucial elements in the proofs of these lower bounds; indeed, our overall argument involves applications of a PCP in two different places. Previously, all concrete circuit lower bounds proved via the algorithmic approach have not required the full power of the PCP theorem [5, 6] for the argument to work.

To formally describe our results, we recall three circuit-analysis problems.

1. **CAPP with error δ:** Given a circuit C on n inputs, estimate the probability that $C(a) = 1$ over uniformly random input $a \in \{0,1\}^n$, to within $\pm\delta$.
2. **SAT:** Given a circuit C, determine if there is an input a such that $C(a) = 1$.
3. **Gap-UNSAT with gap δ:** Given a circuit C, output YES when $C(a) = 0$ for all a, and NO when C has at least $\delta \cdot 2^n$ satisfying assignments. Note that $\mathsf{Gap\text{-}UNSAT}$ with gap δ is easier than the other two problems: either a SAT algorithm or a CAPP algorithm with error δ would immediately imply a $\mathsf{Gap\text{-}UNSAT}$ algorithm with gap δ.

1.1 Equivalence Between Algorithmic Analysis of $\mathsf{THR} \circ \mathsf{THR}$, $\mathsf{THR} \circ \mathsf{MAJ}$, $\mathsf{MAJ} \circ \mathsf{MAJ}$

Our first results give equivalences between algorithmic analysis of $\mathsf{THR} \circ \mathsf{THR}$, $\mathsf{THR} \circ \mathsf{MAJ}$, and $\mathsf{MAJ} \circ \mathsf{MAJ}$ circuits. These equivalences are surprising, because the latter two classes are *provably weaker* than $\mathsf{THR} \circ \mathsf{THR}$ [16]. In fact, $2^{\Omega(n)}$-size lower bounds are well-known for them [23, 19].

Poly-Size THR ∘ MAJ and THR ∘ THR are Equivalent for Circuit-Analysis Algorithms

We say an algorithm running on n-input circuits is *non-trivial* if for all c, it runs in $2^n/n^{\omega(1)}$ time for all circuits of size n^c. We first show that, in terms of designing non-trivial SAT or CAPP algorithms, THR ∘ MAJ and THR ∘ THR are *equally hard or easy.*

▶ **Theorem 1.** *The following two statements hold:*

- ***Equivalence of Non-Trivial SAT Algorithms****: There is a non-trivial SAT algorithm for THR ∘ MAJ circuits of* poly(n)*-size* if and only if *there is such an algorithm for* poly(n)*-size THR ∘ THR circuits.*
- ***Equivalence of Non-Trivial CAPP Algorithms With Constant Error****: For any constant $\delta > 0$, If there is a non-trivial CAPP algorithm with error δ for THR ∘ MAJ circuits of* poly(n) *size, then there is a non-trivial CAPP algorithm with error $\delta + 1/n$ for* poly(n)*-size THR ∘ THR circuits.*

Theorem 1 generalizes readily to TC circuits of any constant depth d. Let LT_d be the class of the depth-d circuits consisting entirely of arbitrary linear threshold functions, and let $\widehat{\mathsf{LT}}_d$ be the subclass of LT_d with the restriction that all gates have polynomially-bounded integer weights (see Section 2.1 for formal definitions). E.g., $\widehat{\mathsf{LT}}_2 = \mathsf{MAJ} \circ \mathsf{MAJ}$.

▶ **Corollary 2.** *The following two statements hold for every constant d:*

- ***Equivalence of Non-Trivial SAT Algorithms****: There is a non-trivial SAT algorithm for THR ∘ $\widehat{LT}_{d-1}$ circuits of* poly(n)*-size* if and only if *there is such an algorithm for* poly(n)*-size LT_d circuits.*
- ***Equivalence of Non-Trivial CAPP Algorithms With Constant Error****: For any constant $\varepsilon > 0$, if there is a non-trivial CAPP algorithm with error ε for THR ∘ $\widehat{LT}_{d-1}$ circuits of* poly(n)*-size, then there is a non-trivial CAPP algorithm with error $\varepsilon + 1/n$ for* poly(n)*-size LT_d circuits.*

Weaker Equivalence Between Poly-Size THR ∘ THR and MAJ ∘ MAJ

We also obtain some weaker equivalences between circuit-analysis algorithms for THR ∘ THR and MAJ ∘ MAJ circuits.

▶ **Theorem 3.** *The following two statements hold:*

- ***Equivalence of $2^{(1-\varepsilon)n}$-time SAT Algorithms****: If SAT for* poly(n)*-size MAJ ∘ MAJ circuits is in $2^{(1-\varepsilon)n}$ time for an $\varepsilon > 0$, then SAT for* poly(n)*-size THR ∘ THR circuits is in $2^{(1-\varepsilon')n}$ time for an $\varepsilon' > 0$.*
- ***Equivalence of Non-Trivial CAPP Algorithms with Inverse Polynomial Error****: If there is a non-trivial CAPP algorithm with $1/poly(n)$ error for* poly(n)*-size MAJ ∘ MAJ circuits, then there is a non-trivial CAPP algorithm with $1/$* poly(n) *error for* poly(n)*-size THR ∘ THR circuits.*

Again, a similar result holds for TC circuits of any constant depth d.

▶ **Corollary 4.** *The following two statements hold for any constant d:*

- ***Equivalence of $2^{(1-\varepsilon)n}$-time SAT Algorithms****: If SAT for $\widehat{LT}_d$ circuits of* poly(n)*-size is in $2^{(1-\varepsilon)n}$ time for an $\varepsilon > 0$, then SAT for* poly(n)*-size LT_d circuits is in $2^{(1-\varepsilon')n}$ time for an $\varepsilon' > 0$.*

- ***Equivalence of Non-trivial CAPP Algorithms with inverse polynomial error:*** *If there is a non-trivial CAPP algorithm with $1/poly(n)$ error for $\mathrm{poly}(n)$-size $\widehat{LT}_d$ circuits, then there is a non-trivial CAPP algorithm with $1/\mathrm{poly}(n)$ error for $\mathrm{poly}(n)$-size LT_d circuits.*

1.2 Tighter Connection Between Circuit Lower Bounds and Non-trivial Derandomization

Our next results give a tighter connection between non-trivial circuit-analysis algorithms for $\mathcal{C}$, and circuit lower bounds against $\mathcal{C}$. We say a circuit class $\mathcal{C}$ is *typical* if it is closed under taking negations of the output, and variable projections.[3] We show that, to prove $\mathsf{NEXP} \not\subset \mathcal{C}$, it suffices to obtain non-trivial derandomization of $\mathsf{AND}_3 \circ \mathcal{C}$, $\mathsf{OR}_2 \circ \mathcal{C}$, or $\oplus_2 \circ \mathcal{C}$ (a.k.a. $\mathsf{XOR}_2 \circ \mathcal{C}$).

▶ **Theorem 5** (Lower Bounds From Non-Trivial Gap-UNSAT or CAPP Algorithms). *There is an absolute constant $\delta > 0$, such that for any typical circuit class $\mathcal{C}$, if one of the following holds:*

- *there is a non-trivial Gap-UNSAT algorithm with gap δ for $\mathrm{poly}(n)$-size $AND_3 \circ \mathcal{C}$ circuits, or*
- *there is a non-trivial CAPP algorithm with error δ for $\mathrm{poly}(n)$-size $OR_2 \circ \mathcal{C}$, $\oplus_2 \circ \mathcal{C}$, or $AND_2 \circ \mathcal{C}$ circuits,*

then $NEXP \not\subset \mathcal{C}$. Moreover, in the second bullet, $\mathcal{C}$ does not need to be closed under negation.

Comparison with Ben-Sasson and Viola

Ben-Sasson and Viola [12] showed that non-trivial Gap-UNSAT algorithms for $\mathsf{OR}_{\mathrm{poly}(n)} \circ \mathsf{AND}_3 \circ \mathcal{C}$, or non-trivial satisfiability algorithms for $\mathsf{AND}_3 \circ \mathcal{C}$ would imply $\mathsf{NEXP} \not\subset \mathcal{C}$. Theorem 5 is a strict strengthening of these two connections, as Gap-UNSAT is an easier problem than SAT. In particular, note we avoid the unbounded fan-in OR entirely.

Applying the "easy witness lemma for NP" results of Murray and Williams [34], we can naturally generalize to circuit lower bounds for NP if faster algorithms are used.

▶ **Theorem 6** (NP Lower Bounds From Faster Gap-UNSAT or CAPP Algorithms). *There is an absolute constant $\alpha > 0$, such that for any typical circuit class $\mathcal{C}$, if there is a constant δ such that one of the following holds:*

- *Gap-UNSAT for $2^{\delta n}$-size $AND_3 \circ \mathcal{C}$ circuits with gap α can be solved in $2^{n-\delta n}$ time, or*
- *CAPP for $2^{\delta n}$-size $OR_2 \circ \mathcal{C}$, $\oplus_2 \circ \mathcal{C}$, or $AND_2 \circ \mathcal{C}$ circuits with error α can be solved in $2^{n-\delta n}$ time,*

then for every k there is a function in NP that doesn't have n^k-size $\mathcal{C}$ circuits. Moreover, in the second bullet, $\mathcal{C}$ does not need to be closed under negation.

▶ Remark 7. In Theorems 5 and 6, the desired algorithms can even be non-deterministic, as long as on all computation paths the algorithm either outputs *don't know* or the correct answer, and the correct answer always appears on at least one path.

In terms of techniques, our approach is very different from that of the previous derandomization connection proved by Ben-Sasson and Viola [12]. They constructed a highly efficient PCP for $\mathsf{NTIME}[T(n)]$, where the queries are *projections* of random bits, and the verifier is a 3-CNF. Their results were then obtained by directly plugging this PCP construction into the original argument of [49].

[3] See Section 2.1 for the details.

Our approach is less direct. Our key insight in proving Theorem 5 (and Theorem 6 similarly) is to use PCPs of Proximity to reduce circuit evaluation tasks to derandomization tasks. Using efficient PCPs for $\mathsf{NTIME}[2^n]$ [10], Williams [49] showed $\mathsf{NEXP} \not\subset \mathsf{P/poly}$ follows from non-trivial Gap-UNSAT algorithms for $\mathrm{poly}(n)$-size general circuits. Applying *PCPs of Proximity* to the *Circuit Evaluation Problem*, we design a Gap-UNSAT algorithm for *general* circuits, only assuming that $\mathsf{NEXP} \subset \mathcal{C}$ and a Gap-UNSAT algorithm for $\mathsf{AND}_3 \circ \mathcal{C}$, which results in a contradiction when $\mathcal{C} \subset \mathsf{P/poly}$. Therefore our overall argument applies PCP constructions in two different ways: first on a nondeterministic 2^n-time computation to reduce to a Gap-UNSAT problem, and then on a $\mathrm{poly}(n)$-size circuit evaluation. See Section 1.6 for an overview of the whole argument.

1.3 Potential Approaches to THR ∘ THR Circuit Lower Bounds

As a direct corollary of Theorem 5 and the folklore result that and XOR of two $\mathsf{THR} \circ \mathsf{THR}$ can be written as a polynomially-larger $\mathsf{THR} \circ \mathsf{THR}$,[4] it follows that non-trivial CAPP algorithms for $\mathsf{THR} \circ \mathsf{THR}$ circuits with small constant error would imply $\mathsf{NEXP} \not\subset \mathsf{THR} \circ \mathsf{THR}$. With a little additional work, the same can be shown for non-trivial SAT algorithms for $\mathsf{THR} \circ \mathsf{THR}$ circuits.

▶ **Theorem 8.** *There is an absolute constant $\delta > 0$, such that if δ-error CAPP for $\mathrm{poly}(n)$-size $\mathsf{THR} \circ \mathsf{THR}$ circuits can be solved in $2^n/n^{\omega(1)}$ time, then $\mathsf{NEXP} \not\subset \mathsf{THR} \circ \mathsf{THR}$. The same is true with SAT in place of CAPP.*

The above theorem generalizes to TC circuits of any constant depth d (LT_d circuits).

▶ **Theorem 9.** *There is an absolute constant $\delta > 0$, such that for any constant d, if CAPP for $\mathrm{poly}(n)$-size LT_d circuits with error δ can be solved in $2^n/n^{\omega(1)}$ time, then $\mathsf{NEXP} \not\subset \mathsf{LT}_d$. The same is true with SAT in place of CAPP.*

It still appears to be a tough challenge to obtain a non-trivial CAPP algorithm for polynomial-size $\mathsf{THR} \circ \mathsf{THR}$ circuits, as it is usually the case that derandomizations come from circuit lower bounds (and ironically, our goal here is to prove circuit lower bounds for $\mathsf{THR} \circ \mathsf{THR}$!). However, armed with our new equivalence results between algorithmic analysis of $\mathsf{THR} \circ \mathsf{THR}$ circuits and $\mathsf{THR} \circ \mathsf{MAJ}$ or $\mathsf{MAJ} \circ \mathsf{MAJ}$ circuits (Theorem 1 and Theorem 3), it suffices for us to obtain non-trivial CAPP algorithms for $\mathsf{THR} \circ \mathsf{MAJ}$ or $\mathsf{MAJ} \circ \mathsf{MAJ}$ circuits, for which $2^{\Omega(n)}$ lower bounds are known.

▶ **Corollary 10.** *There is an absolute constant $\delta > 0$, such that if one of the following holds:*
1. *CAPP (or SAT) for $\mathrm{poly}(n)$-size $\mathsf{THR} \circ \mathsf{MAJ}$ circuits with error δ is in $2^n/n^{\omega(1)}$ time, or*
2. *CAPP for $\mathrm{poly}(n)$-size $\mathsf{MAJ} \circ \mathsf{MAJ}$ circuits with $1/poly(n)$ error is in $2^n/n^{\omega(1)}$ time,*

then $\mathsf{NEXP} \not\subset \mathsf{THR} \circ \mathsf{THR}$.

Therefore $\mathsf{NEXP} \not\subset \mathsf{THR} \circ \mathsf{THR}$ follows if we can "mine" the known $2^{\Omega(n)}$ lower bounds for $\mathsf{THR} \circ \mathsf{MAJ}$ or $\mathsf{MAJ} \circ \mathsf{MAJ}$, and design non-trivial circuit-analysis algorithms from them!

1.4 Lower Bounds on Representing Boolean Functions Approximately by Linear Combinations of Simple Functions

Finally, we apply our new techniques to strengthen recent depth-two lower bounds of R. Williams [48]. He studied the problem of representing a Boolean function f exactly by a linear combination of simple functions from a class $\mathcal{C}$. Here we introduce an approximate form of such representations.

[4] See e.g. Lemma 50 for a proof.

▶ **Definition 11.** *Let $\mathcal{C}$ be a class of functions from $\{0,1\}^n \to \mathbb{R}$ and $\varepsilon \in [0, 0.5)$. We say $f : \{0,1\}^n \to \{0,1\}$ admits a $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathcal{C}$ circuit of sparsity S, if there are S functions $C_1, C_2, \ldots, C_S$ from $\mathcal{C}$, together with S coefficients $\alpha_1, \alpha_2, \ldots, \alpha_S$ in $\mathbb{R}$, such that for all $x \in \{0,1\}^n$,*

$$\left| \sum_{i=1}^{S} \alpha_i \cdot C_i(x) - f(x) \right| \leq \varepsilon.$$

We use $\mathsf{Sum} \circ \mathcal{C}$ to denote the special case of $\varepsilon = 0$, which was the case studied in prior work [48].

When $\mathcal{C}$ is the class of AND gates (or PARITY gates, respectively), we are asking for the *sparsest* ε-approximate polynomial for f, with respect to the standard (or Fourier basis, respectively). This is related to the ε-approximate degree[5] of f, which is already a highly-nontrivial notion; for instance, the approximate degrees of simple natural functions have only recently been determined [15, 14, 42].

In prior work, Williams [48] showed that non-trivial algorithms for the so-called "Sum-Product"[6] of $O(1)$ functions from $\mathcal{C}$ implies sparsity lower bounds against $\mathsf{Sum} \circ \mathcal{C}$, and he obtained sparsity lower bounds against various $\mathsf{Sum} \circ \mathcal{C}$ circuits by designing corresponding Sum-Product algorithms.

Applying our new techniques together with other new ideas, we show that Sum-Product algorithms in fact yield sparsity lower bounds against $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathcal{C}$. That is, we can systematically "lift" the $\mathsf{Sum} \circ \mathcal{C}$ lower bounds in [48] to lower bounds for $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathcal{C}$.

First, we generalize the lower bounds for $\mathsf{Sum} \circ \mathsf{THR}$ in [48] to $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathsf{THR}$. Such circuits are also known in the machine learning literature as *depth-two neural networks with sign activation functions*.

▶ **Theorem 12** (Lower Bound for $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathsf{THR}$)**.** *For all k and constant $\varepsilon < 1/2$, there is a function in NP without $\widetilde{\mathsf{Sum}}_\varepsilon \circ$ THR circuits of n^k sparsity. Furthermore, if $\alpha(n)$ is unbounded such that $n^{\alpha(n)}$ is time-constructible, then $\mathsf{NTIME}[n^{\alpha(n)}] \not\subset \widetilde{\mathsf{Sum}}_\varepsilon \circ$ THR for all constant $\varepsilon < 1/2$.*

A ReLU (rectified linear unit) gate is a function $f : \{0,1\}^t \to \mathbb{R}^+$ such that there is a vector $w \in \mathbb{R}^t$ and scalar $a \in \mathbb{R}$ such that for all x,

$$f(x) = \max\{0, \langle x, w \rangle + a\}.$$

Linear combinations of ReLU gates are also known as *depth-two neural networks with ReLU activation functions*, and they are intensely studied in machine learning.[7]

Next we generalize the lower bounds for $\mathsf{Sum} \circ \mathsf{ReLU}$ in [48] to $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathsf{ReLU}$.

▶ **Theorem 13** (Lower Bound for $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathsf{ReLU}$)**.** *For all k and constant $\varepsilon < 1/2$, there is a function in NP without $\widetilde{\mathsf{Sum}}_\varepsilon \circ$ ReLU circuits of n^k sparsity. Furthermore, if $\alpha(n)$ be unbounded such that $n^{\alpha(n)}$ is time-constructible, then $\mathsf{NTIME}[n^{\alpha(n)}] \not\subset \widetilde{\mathsf{Sum}}_\varepsilon \circ$ ReLU for all constant $\varepsilon < 1/2$.*

We also obtain analogous lower bounds for $\widetilde{\mathsf{Sum}}_\varepsilon \circ (O(1)$-degree $\mathbb{F}_p$-polynomials), strengthening lower bounds for exact linear combinations of $O(1)$-degree polynomials [48].

---

[5] The ε-approximate degree of f is the lowest degree of all polynomial $p : \{0,1\}^n \to \{0,1\}$ such that $\|p - f\|_\infty \leq \varepsilon$. Note that a low degree polynomial is also sparse.

[6] See Section 6 for a formal definition. Intuitively, the "Sum-Product" problem generalizes #SAT.

[7] We refer the readers to [48] and the references therein for more discussion on this topic.

▶ **Theorem 14** (Lower Bound for $\widetilde{\mathsf{Sum}}_\varepsilon \circ (\mathbb{F}_p\text{-polynomials})$). *For all prime p, integers k, d, and constant $\varepsilon < 1/2$, there is a function in NP without $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathsf{MOD}_p \circ \mathsf{AND}_d$ circuits of n^k sparsity. Furthermore, if $\alpha(n)$ be unbounded such that $n^{\alpha(n)}$ is time-constructible, then $\mathsf{NTIME}[n^{\alpha(n)}] \not\subset \widetilde{\mathsf{Sum}}_\varepsilon \circ \mathsf{MOD}_p \circ \mathsf{AND}_d$ for all constant $\varepsilon < 1/2$.*

Finally, using the known #SAT algorithm for $\mathsf{ACC}^0 \circ \mathsf{THR}$ [51], we show a sparsity lower bound for $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathsf{ACC}^0 \circ \mathsf{THR}$ circuits.

▶ **Theorem 15.** *For every $d, m \geq 1$ and $\varepsilon \in [0, 0.5)$, there is a $b \geq 1$ and an $f \in \mathsf{NTIME}[n^{\log^b n}]$ that does not have $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathsf{AC}^0_d[m] \circ \mathsf{THR}$ circuits of n^a size, for every a.*

Therefore, no polynomially-sparse linear combination of $\mathsf{AC}^0_d[m] \circ \mathsf{THR}$ circuits can approximate the value of the hard function in Theorem 15.

This constitutes the strongest known circuit class for which we can presently prove lower bounds for nondeterministic quasi-polynomial time (improving [34]).

1.5 Techniques: Two Structure Lemmas for THR ∘ THR

Two major technical ingredients in our results are structure lemmas for $\mathsf{THR} \circ \mathsf{THR}$, which are of interest in their own right. Informally, our first structure lemma says that every $\mathsf{THR} \circ \mathsf{THR}$ is equivalent to a polynomial-sized OR of Threshold-of-Majority circuits. The second structure lemma says that every $\mathsf{THR} \circ \mathsf{THR}$ circuit is equivalent to a subexponential-sized OR of Majority-of-Majority circuits. For the program of proving $\mathsf{THR} \circ \mathsf{THR}$ lower bounds, this is significant, as exponential-size Majority-of-Majority and Threshold-of-Majority lower bounds are well-known [23, 19].

In the following, DOR refers to a "disjoint" OR gate: an OR gate with the promise that at most one of its inputs is ever true, and $\mathsf{Gap\text{-}OR}_\delta$ refers to a "gapped" OR gate with a error parameter δ: an OR gate with the promise that either all inputs are false or at least a $1 - \delta$ fraction of the inputs are true. We also use Gap-OR to denote $\mathsf{Gap\text{-}OR}_{1/2}$ for simplicity. (See Section 2.1 for formal definitions.)

▶ **Lemma 16** (Structure Lemma I for $\mathsf{THR} \circ \mathsf{THR}$ circuits). *Let n be the number of inputs, let $s = s(n) \geq n$ be a size function, and let $\delta = \delta(n) \in (0, 1)$ be an error function. Every s-size $\mathsf{THR} \circ \mathsf{THR}$ circuit C is equivalent to a $\mathsf{Gap\text{-}OR}_\delta \circ \mathsf{THR} \circ \mathsf{MAJ}$ circuit C' such that:*

- *The top $\mathsf{Gap\text{-}OR}_\delta$ gate of C' has $\mathrm{poly}(s, \delta^{-1})$ fan-in.*
- *Each $\mathsf{THR} \circ \mathsf{MAJ}$ subcircuit of C' has size $\mathrm{poly}(s, \delta^{-1})$.*

The transformation from C to C' can be computed in deterministic $\mathrm{poly}(s, \delta^{-1})$ time.

▶ **Lemma 17** (Structure Lemma II for $\mathsf{THR} \circ \mathsf{THR}$ circuits). *Let n be the number of inputs and let $s = s(n) \leq 2^{o(n)}$ be a size parameter. Let $\varepsilon \in \left(\frac{\log s}{n}, 1\right)$. Every s-size $\mathsf{THR} \circ \mathsf{THR}$ circuit C is equivalent to a $\mathsf{DOR} \circ \mathsf{MAJ} \circ \mathsf{MAJ}$ circuit such that:*

- *The top DOR gate has $2^{O(\varepsilon n)}$ fan-in.*
- *Each sub $\mathsf{MAJ} \circ \mathsf{MAJ}$ circuit has size $s^{O(1/\varepsilon)}$.*

The reduction can be computed in randomized $2^{O(\varepsilon n)} \cdot s^{O(1/\varepsilon)}$ *time.*

Previously, Goldmann-Håstad-Razborov [21] showed that every $\mathsf{THR} \circ \mathsf{THR}$ circuit has an equivalent $\mathsf{MAJ} \circ \mathsf{MAJ} \circ \mathsf{MAJ}$ circuit of polynomially larger size. The top OR gates in our structure lemmas have additional benefits: for instance, an $\mathsf{OR} \circ \mathcal{C}$ circuit is satisfiable, *if and only if* one of its $\mathcal{C}$ subcircuits is satisfiable. Therefore, solving SAT on an $\mathsf{OR} \circ \mathcal{C}$ circuit is easily reduced to solving SAT on $\mathcal{C}$ circuits.

In Appendix C, we discuss more applications of the above two structure lemmas, beyond the algorithmic equivalences for $\mathsf{THR} \circ \mathsf{THR}$, $\mathsf{THR} \circ \mathsf{MAJ}$, and $\mathsf{MAJ} \circ \mathsf{MAJ}$ circuits.

1.6 Intuition: Solving Gap-UNSAT with Probabilistic Checkable Proofs of Proximity

Here we provide an overview of the ideas behind our new tightened connection between circuit lower bounds and circuit-analysis algorithms.

Starting Point: Designing Gap-UNSAT Algorithms for General Circuits, Assuming NEXP $\subset \mathcal{C}$

Suppose $\mathcal{C} \subset \mathsf{P}/\mathrm{poly}$. We want to show that a non-trivial Gap-UNSAT algorithm with a constant gap for poly(n)-size $\mathsf{AND}_3 \circ \mathcal{C}$ circuits implies $\mathsf{NEXP} \not\subset \mathcal{C}$. We start with the following connection of R. Williams [49]:

> If Gap-UNSAT with gap $1-1/n^{10}$ for (fan-in 2) circuits with n inputs and poly(n) size is solvable in $O(2^n/n^{\omega(1)})$ nondeterministic time, then NEXP doesn't have poly(n)-size (fan-in 2) circuits.

Our strategy is to assume $\mathsf{NEXP} \subset \mathcal{C}$, and use our non-trivial Gap-UNSAT algorithm for $\mathsf{AND}_3 \circ \mathcal{C}$ to derive a non-trivial Gap-UNSAT algorithm for *general* fan-in-2 circuits. This would imply a contradiction, since by the above connection, it follows that $\mathsf{NEXP} \not\subset \mathsf{P}_{/\mathrm{poly}}$ and therefore $\mathsf{NEXP} \not\subset \mathcal{C}$.

So suppose we are given a poly(n)-size general circuit $C : \{0,1\}^n \to \{0,1\}$ with the promise that either C is unsatisfiable (the YES case) or C has at least $(1 - 1/n^{10}) \cdot 2^n$ satisfying assignments (the NO case), where our goal is to distinguish the two cases in $2^n/n^{\omega(1)}$ non-deterministic time.

To simplify the discussion, we negate the circuit C. Now we are promised C is a tautology, or C has at most $1/n^{10} \cdot 2^n$ satisfying assignments, and we must nondeterministically prove C is a tautology (when that is the case) in $2^n/n^{\omega(1)}$ time.

Review of the Approach in Williams' ACC Lower Bound

It will be useful to review the previous approach ([52]) first, and see where we deviate from it.[8] Let the circuit C be given as above. First, assuming $\mathsf{NEXP} \subset \mathcal{C}$ (which implies Circuit-Eval $\in \mathcal{C}$), there is an equivalent poly(n)-size $\mathcal{C}$ circuit D equivalent to C. Since we are allowed to use *non-deterministic* algorithms, we might try to *guess* a $\mathcal{C}$ circuit D, and *verify* that D is equivalent to C. If this verification can be done in $2^n/n^{\omega(1)}$ time, then we could apply the Gap-UNSAT algorithm for $\mathcal{C}$ to the circuit D, and solve Gap-UNSAT for C. Indeed, this is the original approach of Wiliams [52].

Since the NAND gate ($\mathsf{NAND}(z_1, z_2) := \neg(z_1 \wedge z_2)$) is universal, we may assume C consists of $m = $ poly(n) NAND gates, the first n gates are the inputs (that is, the i-th gate is the input bit x_i for $i \in [n]$), and the m-th gate is the output gate. Let C_i be the subcircuit of C where the i-th gate is the output. Since we are assuming Circuit-Eval $\in \mathcal{C}$, for all C_i there is always an equivalent $\mathcal{C}$-circuit T_i of poly(n) size.

The overall guess-and-verify algorithm works as follows:

- Guess $m - n$ $\mathcal{C}$-circuits $T_{n+1}, T_{n+2}, \ldots, T_m$, such that T_i is intended to be equivalent to C_i. For $i \in [n]$, we set T_i to be a trivial circuit which always outputs the i-th bit of the input.

[8] Our presentation here is slightly different from the original proof.

- For $i \in \{n+1, n+2, \ldots, m\}$, let i_1 and i_2 be the indices of the two gates which are inputs to the i-th gate of C. We want to verify

$$\mathsf{NAND}(T_{i_1}(x), T_{i_2}(x)) = T_i(x) \tag{1}$$

  is true for all $x \in \{0,1\}^n$. This can be reduced to solving SAT for $\mathsf{AND}_3 \circ \mathcal{C}$ circuits.
- If all the above checks pass, then we know T_m is equivalent to C.

The Proof System View

The above approach requires using SAT algorithms to verify (1) is true for all $x \in \{0,1\}^n$, whereas we only want to assume non-trivial $\mathsf{Gap\text{-}UNSAT}$ algorithms (which could be much weaker). Here we present a different perspective on the above approach.

Letting $\pi(x) := (T_{n+1}(x), T_{n+2}(x), \ldots, T_m(x))$, we can view $\pi(x)$ as a certain "locally-checkable proof" for $C(x) = 1$. That is, $C(x) = 1$ if and only if there is a proof $\pi(x) \in \{0,1\}^{m-n}$ such that for the string $z = x \circ \pi(x)$ ($\circ$ means concatenation), we have $\mathsf{NAND}(z_{i_1}, z_{i_2}) = z_i$ for all $i \in \{n+1, n+2, \ldots, m\}$, and $z_m = 1$.

Can we obtain something better from the "locally-checkable" perspective? We may write all the constraints checked in our proof system as a 3-CNF formula φ on $z = x \circ \pi(x)$ of $\ell = O(m) = \mathrm{poly}(n)$ clauses. (Note, this simply mimics the standard reduction from $\mathsf{Circuit\text{-}Eval}$ to 3-SAT.) Suppose the i-th clause is $F_i(z) := \vee_{j=1}^{3}(z_{i_j} \oplus b_{i,j})$.

- As before, we guess $\mathcal{C}$ circuits $T_{n+1}(x), T_{n+2}(x), \ldots, T_m(x)$, but this time with the intention that $T(x) = (T_{n+1}(x), T_{n+2}(x), \ldots, T_m(x))$ is the correct proof for input x.
- When C is a tautology, there is a guess $T(x)$ such that $\mathbb{E}_{x \sim \mathcal{U}_n} \mathbb{E}_{i \in [\ell]}[F_i(x \circ T(x))] = 1$.
- Otherwise, for all guessed $T(x)$, we have $\mathbb{E}_{x \sim \mathcal{U}_n} \mathbb{E}_{i \in [\ell]}[F_i(x \circ T(x))] \leq 1/n^{10} + \frac{\ell-1}{\ell}$, since for at least a $1 - 1/n^{10}$ fraction of inputs, we have $C(x) = 0$, and therefore at most $\ell - 1$ clauses can be satisfied by $x \circ T(x)$.

Note that $F_i(x \circ T(x))$ is an $\mathsf{OR}_3 \circ \mathcal{C}$ circuit. We can try to estimate $\mathbb{E}_{x \sim \mathcal{U}_n}[F_i(x \circ T(x))]$ for each $i \in [\ell]$ to distinguish between the above two cases. Note there is only a $1/\ell = 1/\mathrm{poly}(n)$ gap between the above two cases. Therefore, this argument does show that, if we assume to have non-trivial CAPP algorithms for $\mathsf{OR}_3 \circ \mathcal{C}$ with $1/\mathrm{poly}(n)$ error, the above guess-and-verify approach already suffices to obtain lower bounds against $\mathcal{C}$.

However, in our case, we are only assuming to have a $\mathsf{Gap\text{-}UNSAT}$ algorithm with a *constant* gap. It is not clear how to make further progress with the above idea.

A Better Proof System?

The above idea does not work, essentially because the described "proof system" is a pretty *bad* PCP! Given the pair $(x, T(x))$, if the verifier draws a random $i \in [\ell]$ and checks whether the clause F_i is satisfied, it is only promised to detect an error with probability $\geq 1/\ell$ when $C(x) = 0$ and the proof $T(x)$ is incorrect. In other words, it has a completeness/soundness gap of only $1/\ell = 1/\mathrm{poly}(n)$. A natural response to this observation is to try using a better proof system for proving that $C(x) = 1$; it comes as no surprise that we turn to the PCP Theorem [5, 6].

However, there is a subtle issue. In the above proof system, the verifier does not need to know the input x beforehand, and only needs to query a bit of x when verifying a clause F_i containing that bit. The most important property here is that *the verifier's queries do not depend on the input* x, as otherwise we cannot formulate the condition "the verifier accepts with the random index i and proof $T(x)$ on input x" as a simple function $F_i(x \circ T(x))$ which can be represented by an $\mathsf{OR}_3 \circ \mathcal{C}$ circuit.

Suppose we forced the verifier to access the input x using only $O(1)$ queries, as in the above proof system, but the circuit is computing a highly-sensitive function such as the parity of x. There is no way that a verifier querying x for only $O(1)$ times can correctly infer (with high probability) that the parity of x is odd! This is because if the parity of x is odd, the parity will change if we flip a random bit of x, so it is not possible for a verifier to distinguish between these two cases with constant probability, if the verifier can only query x for $O(1)$ times.

Error Correcting Codes and Probabilistic Checkable Proofs of Proximity

To avoid the above trivial counterexample, our next key idea is to provide the PCP verifier an *error-correcting encoding* of the input. Now we are at the right position to introduce the main technical concept used in this paper: *Probabilistic Checkable Proofs of Proximity* (PCPP) for the Circuit-Eval problem. When properly applied, PCPPs allow us to reduce the error requirement on the CAPP algorithms from *inverse polynomial* to only a *constant*.

In this type of proof system[9], a circuit E is fixed in advance, the verifier $V(E)$ gets oracle access to the input x of length n and a proof string π, tosses some random coins, and makes at most 3 *non-adaptive* queries. The proof system has constant parameters $\delta > 0$ and $s \in (0, 1)$, and satisfies two important properties:

- (Perfect Completeness.) $E(x) = 1 \Rightarrow$ there is a π such that $\Pr[V(E) \text{ accepts } x \circ \pi] = 1$.
- (Soundness on inputs far from being correct.) If x is δ-far from the set $\{y : E(y) = 1\}$, where δ is the proximity parameter, then for all possible proofs π, $V(E)$ accepts $x \circ \pi$ with probability at most $s < 1$.

To clarify the second point, we are saying that if x has hamming distance more than δn from all y that satisfy E, then $V(E)$ has decent probability of rejection on *any* proof π.

Suppose we use a linear error correcting code with an efficient encoder Enc and decoder Dec, and define the circuit E by $E(y) := C(\mathsf{Dec}(y))$. That is, E treats its input y as an encoding of an input to the circuit C; it first decodes y to a string z, then feeds z to C to get its output.

Let $x \in \{0, 1\}^n$ be an input to C. We instantiate a PCP of proximity proof system with the circuit E and the input $\mathsf{Enc}(x)$. It is not hard to see that when $C(x) = 0$, $\mathsf{Enc}(x)$ is δ_1-far from the accepting inputs for E for a constant δ_1 depending on the error correcting code. We can ensure that $\delta_1 > \delta$.

The Final Reduction

Now, suppose there are ℓ possible outcomes of the random coins, and assume that the proof π is of length ℓ as well. Let $F_i(\mathsf{Enc}(x) \circ T(x))$ be the indicator that given a random outcome $i \in [\ell]$, whether the verifier $V(E)$ accepts the oracle $\mathsf{Enc}(x) \circ T(x)$. By definition, $F_i(\mathsf{Enc}(x) \circ T(x))$ is a function on 3 coordinates of $\mathsf{Enc}(x) \circ T(x)$ (we can assume WLOG that F_i is simply an OR, by using a special PCP of proximity proof system; see Lemma 24). Note that a bit of $\mathsf{Enc}(x)$ is just a parity over a subset of bits in x. For simplicity, let us further assume $\mathcal{C} = \mathsf{THR} \circ \mathsf{THR}$, which can compute parity (note, this assumption can be removed). Then $F_i(\mathsf{Enc}(x) \circ T(x))$ can now be formulated as an $\mathsf{OR}_3 \circ \mathcal{C}$ circuit. Now we proceed similarly as before.

[9] see Definition 20

- We again try to guess $\mathcal{C}$ circuits $T_1(x), T_1(x), \ldots, T_\ell(x)$, but this time with the hope that $T(x) = (T_1(x), T_2(x), \ldots, T_\ell(x))$ is the correct proof for the verifier $V(E)$ given input $\mathsf{Enc}(x)$.
- When C is a tautology, there is a guess $T(x)$ such that $\mathbb{E}_{x\sim\mathcal{U}_n}\mathbb{E}_{i\in[\ell]}[F_i(\mathsf{Enc}(x)\circ T(x))] = 1$.
- Otherwise, for all guesses $T(x)$, $\mathbb{E}_{x\sim\mathcal{U}_n}\mathbb{E}_{i\in[\ell]}[F_i(\mathsf{Enc}(x)\circ T(x))] \leq 1/n^{10} + s$, since for at least a $1 - 1/n^{10}$ fraction of inputs, we have $C(x) = 0$, and therefore at most an s fraction of F_i's can be satisfied by $\mathsf{Enc}(x) \circ T(x)$, because $\mathsf{Enc}(x)$ is δ-far from any accepting input to E.

In this new situation, it now suffices to estimate $\mathbb{E}_{x\sim\mathcal{U}_n}[F_i(x \circ T(x))]$ for each $i \in [\ell]$ within sufficiently small constant error. A careful examination of the above argument shows it suffices to use a non-trivial Gap-UNSAT algorithm for $\mathsf{AND}_3 \circ \mathcal{C}$ circuits with a constant gap (note that the negation of F_i is an $\mathsf{AND}_3 \circ \mathcal{C}$ circuit), because we have perfect completeness in the case where C is a tautology.

Lower Bounds From CAPP Algorithms for $\mathsf{OR}_2 \circ \mathcal{C}$, $\mathsf{AND}_2 \circ \mathcal{C}$, or $\oplus_2 \circ \mathcal{C}$ Circuits

The above shows how to use a non-trivial CAPP algorithm for $\mathsf{OR}_3 \circ \mathcal{C}$; how can we use a non-trivial CAPP algorithm for $\mathsf{OR}_2 \circ \mathcal{C}$, $\mathsf{AND}_2 \circ \mathcal{C}$, or $\oplus_2 \circ \mathcal{C}$? The natural idea is to instead use a 2-query PCPP for Circuit-Eval. Unfortunately, there is no PCPP with only 2 queries with perfect completeness for Circuit-Eval, unless $\mathsf{P} = \mathsf{NP}$.[10] Thus we must use a construction with imperfect completeness. Luckily, there is a 2-query PCPP for Circuit-Eval with a constant soundness/completeness gap (Lemma 25). We use that PCPP in the above argument, together with other ideas, to establish the connection with a non-trivial CAPP algorithm for $\mathsf{OR}_2 \circ \mathcal{C}$, $\mathsf{AND}_2 \circ \mathcal{C}$ or $\oplus_2 \circ \mathcal{C}$ circuits.

1.7 Related Work

For more history on previous works on lower bounds for constant-depth threshold circuits, see the corresponding sections in [51, 31]. We only discuss a few recent results here.

In 2014, Williams [51] showed that NEXP is not contained in $\mathsf{ACC}^0 \circ \mathsf{THR}$, by devising a fast satisfiability algorithm for $\mathsf{ACC}^0 \circ \mathsf{THR}$. The lower bound was recently improved by Murray and Williams [34] to show $\mathsf{NTIME}[n^{\mathrm{polylog}(n)}]$ is not contained in $\mathsf{ACC}^0 \circ \mathsf{THR}$. Tamaki [44], Alman, Chan and Williams [3] proved that E^{NP} does not have $n^{2-o(1)}$ size $\mathsf{THR} \circ \mathsf{THR}$ circuits. Most recently, Williams [48] showed that there are functions in $\mathsf{NTIME}[n^{\log^{\omega(1)}(n)}]$ that can not be represented by a linear combination of polynomially many $\mathsf{ACC}^0 \circ \mathsf{THR}$ circuits.

Tell [46] constructed a *quantified derandomization* algorithm for TC circuits with depth d and $n^{1+\exp(-d)}$ wires, and showed that a modest improvement of his algorithm would imply standard derandomization of TC^0, and consequently $\mathsf{NEXP} \not\subseteq \mathsf{TC}^0$.

Using random restrictions, Kane and Williams [31] proved that any $\mathsf{THR} \circ \mathsf{THR}$ circuits computing Andreev's function requires $\widetilde{\Omega}(n^{1.5})$ gates and $\widetilde{\Omega}(n^{2.5})$ wires. Chattopadhyay and Mande [16] recently showed an exponential size separation between $\mathsf{THR} \circ \mathsf{MAJ}$ and $\mathsf{THR} \circ \mathsf{THR}$, by constructing a function in $\mathsf{THR} \circ \mathsf{THR}$ with exponential *sign-rank*.

[10] A 2-query PCPP for Circuit-Eval with perfect completeness implies a 2-query PCP for NP with perfect completeness [11], which in turn implies $\mathsf{P} = \mathsf{NP}$, as 2-SAT is in P.

1.8 Organization of the Paper

In Section 2 we discuss necessary preliminaries. In Section 3 we prove equivalences between non-trivial circuit analysis tasks of $\mathsf{THR} \circ \mathsf{THR}$ and that of $\mathsf{THR} \circ \mathsf{MAJ}$ or $\mathsf{MAJ} \circ \mathsf{MAJ}$. In Section 4 we establish a tighter connection between non-trivial derandomization and circuit lower bounds. In Section 5, we propose approaches toward proving $\mathsf{NEXP} \not\subset \mathsf{THR} \circ \mathsf{THR}$. In Section 6 we prove lower bounds for various $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathcal{C}$ circuits. In Section 7 we prove two new structure lemmas for $\mathsf{THR} \circ \mathsf{THR}$ circuits.

2 Preliminaries

The *Circuit Evaluation Problem* (Circuit-Eval) is the language of pairs $\{(C, w)\}$ such that when C is a general fan-in-2 circuit and w is a Boolean input, $(C, w) \in$ Circuit-Eval if and only if $C(w) = 1$. For two strings a, b, we use $a \circ b$ to denote their concatenation[11].

2.1 Circuit Classes

Let $\mathcal{C}$ be a circuit class. We use $\mathcal{C}_n^s$ to denote the set of $\mathcal{C}$ of circuits with n inputs and size at most s. Slightly abusing notation, we also use $\mathcal{C}_n^s$ to denote the *Boolean functions* computed by circuits in $\mathcal{C}_n^s$, when convenient.

We say a circuit class $\mathcal{C}$ is *typical*, if given the description of a circuit C from $\mathcal{C}_n^s$, for all indices $1 \leq i, j \leq n$ and $b \in \{0, 1\}$, the following functions are in $\mathcal{C}_n^s$:

$$\neg C,\ C(x_1, \ldots, x_{i-1}, x_j \oplus b, x_{i+1}, \ldots, x_n),\ C(x_1, \ldots, x_{i-1}, b, x_{i+1}, \ldots, x_n).$$

Furthermore, we require that given a description of C, descriptions of all the above circuits can be constructed in $\mathrm{poly}(s)$ time. That is, $\mathcal{C}$ is typical if it is closed under both efficiently-computable *negation* and efficiently-computable *projection*.

Notations for Circuit Classes

As many circuit classes are discussed in this work, we begin with some notation for such classes.

Let $x \in \{0, 1\}^n$. For $w \in \mathbb{R}^n$ and $t \in \mathbb{R}$, we define $\mathsf{THR}_{w,t}(x)$ (the threshold function) to be the indicator function for the condition $w \cdot x \geq t$. Similarly, $\mathsf{ETHR}_{w,t}(x)$ (the exact threshold function) is the indicator function for the condition $w \cdot x = t$. The values in the vector w are called the *weights*, and the real t is called the *threshold* of $\mathsf{THR}_{w,t}$ and $\mathsf{ETHR}_{w,t}$. We say these weights and thresholds are *realizations* of the Boolean functions they define. A fixed Boolean function may have many different realizations. It is known that, without loss of generality, the weights and thresholds are integers of absolute value at most $2^{O(n \log n)}$ [33, 7]. For a threshold or exact threshold function with weight w, we call $L(x) := w \cdot x$ its *associated linear function*.

We use MAJ_n and EMAJ_n to denote the corresponding threshold (exact threshold) functions on n inputs where all weights are 1 and the threshold value is $n/2$. Slightly abusing notation, we also use $\mathsf{THR}, \mathsf{ETHR}, \mathsf{MAJ}, \mathsf{EMAJ}$ to denote the classes of all such functions. We also consider $\oplus_k$ (PARITY), AND_n, and OR_n, with their usual meanings. We use DOR_n to denote the disjoint OR function, that is, an OR function with the promise that at most

[11] Note that we also use $\circ$ for the composition of two circuits, but throughout the paper the meaning of the symbol $\circ$ will always be clear from the context

one input bit is true over all inputs. We use $\mathsf{Gap\text{-}OR}_{n,\delta}$ to denote the gap-OR function on n inputs, that is, an OR function with the promise that either all n inputs are false, or at least a $1-\delta$ fraction of the n inputs are true. (The function may have undefined behavior on other inputs.)

For two classes of functions like THR and MAJ, we use $\mathsf{THR} \circ \mathsf{MAJ}$ to denote the corresponding class of depth-two circuits. Similar notations are used for more than two classes.

We use $\mathsf{AC}^0[m]_d$ to denote depth-d $\mathsf{AC}^0[m]$ circuits (with unbounded fan-in OR, AND, and MOD_m gates). We use LT_d to denote the depth-d THR circuit class, that is, $\mathsf{LT}_d := \underbrace{\mathsf{THR} \circ \ldots \circ \mathsf{THR}}_{d \text{ times}}$. Similarly, we use $\widehat{\mathsf{LT}}_d$ to denote its unweighted version, that is, $\widehat{\mathsf{LT}}_d := \underbrace{\mathsf{MAJ} \circ \ldots \circ \mathsf{MAJ}}_{d \text{ times}}$.

Previous Known Containment Results

We need the following known circuit classes containment results for this paper.

▶ **Proposition 18.** *The following hold:*

1. $\mathsf{THR} \subseteq \mathsf{MAJ} \circ \mathsf{MAJ}$ *[21, 27].*
2. $\mathsf{THR} \subseteq \mathsf{DOR} \circ \mathsf{ETHR}$ *[24]. (also see Appendix B)*
3. $\mathsf{MAJ} \circ \mathsf{THR}$ *and* $\mathsf{MAJ} \circ \mathsf{ETHR}$ *are contained in* $\mathsf{MAJ} \circ \mathsf{MAJ}$ *[21, 24].*
4. $\mathsf{ETHR} \circ \mathsf{ETHR} \subseteq \mathsf{THR} \circ \mathsf{THR}$ *[24].*
5. $\mathsf{AND} \circ \mathsf{ETHR} \subseteq \mathsf{ETHR}$ *[24].*
6. $\mathsf{EMAJ} \subseteq \mathsf{MAJ} \circ \mathsf{AND}_2$ *[24].*
7. $\oplus_k \circ \mathsf{THR} \circ \mathsf{THR} \subseteq \mathsf{THR} \circ \mathsf{THR}$ *for a constant* k*. (see Appendix B)*
8. $\mathsf{THR} \circ \mathsf{EMAJ} \subseteq \mathsf{THR} \circ \mathsf{MAJ}$ *[24].*

Moreover, all the above have corresponding polynomial-time, deterministic constructions.

For the containment $\mathsf{THR} \subseteq \mathsf{DOR} \circ \mathsf{ETHR}$, we present an alternative proof in Appendix B, which is more efficient than the previously known construction of Hansen and Podolskii [24].[12] The last containment is folklore; we present a proof in Appendix B for completeness.

2.2 Approximation Theory

We need the following standard result from approximation theory.

▶ **Lemma 19** ([39] Corollary.1.4.1)**.** *Let* $0 < \varepsilon_1 < \varepsilon_2 < 1/2$ *be two constants, there is an* $O_{\varepsilon_1,\varepsilon_2}(1)$ *degree polynomial* $P : \mathbb{R} \to \mathbb{R}$*, such that:*

- *for all* $z \in [-\varepsilon_2, \varepsilon_2]$*,* $P(z) \in [-\varepsilon_1, \varepsilon_1]$*, and*
- *for all* $z \in [1-\varepsilon_2, 1+\varepsilon_2]$*,* $P(z) \in [1-\varepsilon_1, 1+\varepsilon_1]$*.*

2.3 Probabilistic Checkable Proofs of Proximity

The concept of probabilistically checkable proofs of proximity is crucial for this paper. In the following we introduce its definition and several instantiations useful for this paper.

[12] Hansen and Podolskii [24] proved that a THR gate on n bits with weights of absolute value no greater than W, can be written as a DOR of $O(n^2 \cdot \log W)$ many ETHR gates. In Appendix B we show it can be improved to $O(n \cdot \log W)$ many ETHR gates.

▶ **Definition 20** (Probabilistic Checkable Proofs of Proximity (PCP of proximity, or PCPP)). *For $s, \delta : \mathbb{N} \to [0,1]$ and $r, q : \mathbb{N} \to \mathbb{N}$, a verifier V is a PCP of proximity system for a pair language L with proximity parameter δ, soundness parameter s, number of random bits r and query complexity q if the following holds for all x, y:*

- *If $(x, y) \in L$, there is a proof π such that $V(x)$ accepts oracle $y \circ \pi$ with probability 1.*
- *If y is $\delta(|x|)$-far from $L(x) := \{z : (x, z) \in L\}$, then for all proofs π, $V(x)$ accepts oracle $y \circ \pi$ with probability at most $s(|x|)$.*
- *$V(x)$ tosses $r(|x|)$ random coins, and makes at most $q(|x|)$ non-adaptive queries.*

▶ Remark 21. We can also relax the first condition to be that there is a proof π such that $V(x)$ accepts oracle $y \circ \pi$ with probability at least $c = c(|x|)$, where c is the completeness parameter. In the above definition we assume $c = 1$, i.e., the perfect completeness.

▶ **Lemma 22** (Theorem 3.3 in [11]). *For any constants $0 < \delta, s < 1$, there is a PCP of proximity system for Circuit-Eval with proximity δ, soundness s, number of random bits $r = O(\log n)$ and query complexity $q = O(1)$. Moreover, given the pair $(C, w) \in$ Circuit-Eval, a proof π making $V(C)$ always accepts can be constructed in $\mathrm{poly}(|C| + |w|)$ time.*

▶ Remark 23. The moreover part is not explicitly stated in [11], but it is evident from the constructions.

The exact number of queries used in a PCPP will be significant for us, so we use query-efficient PCPPs. They are already implicit in the literature; for completeness, we provide expositions for them in Appendix A.

▶ **Lemma 24** (3-query PCPP with perfect completeness). *For any constant $\delta > 0$ there is a constant $0 < s < 1$, such that there is a PCP of proximity system for Circuit-Eval with proximity δ, soundness s, random bits $r = O(\log n)$, and query complexity $q = 3$. Moreover, the system satisfies two additional properties:*

(1) *Given the random coins, the verifier simply computes an OR on these 3 queried bits or their negations, and accepts iff the OR is true.*

(2) *Given the pair $(C, w) \in$ Circuit-Eval, we can construct a proof π in $\mathrm{poly}(|C| + |w|)$ time that makes $V(C)$ accept with probability 1.*

▶ **Lemma 25** (2-query PCPP with constant completeness/soundness gap). *For any constant $\delta > 0$ there two constants $0 < s < c < 1$, such that there is a PCP of proximity system for Circuit-Eval with proximity δ, soundness s, completeness c, number of random bits $r = O(\log n)$ and query complexity $q = 2$. Moreover, it satisfies two additional properties:*

(1) *Given the random coins, the verifier computes an OR on the 2 queried bits or their negations, and accepts iff the OR is true.*

(2) *Given the pair $(C, w) \in$ Circuit-Eval, a proof π can be constructed in $\mathrm{poly}(|C| + |w|)$ time that makes $V(C)$ accept with probability at least c.*

2.4 Error Correcting Codes

We also need standard constructions of constant-rate linear error correcting codes.

▶ **Lemma 26** ([43]). *There is a constant $\delta > 0$ such that there is a constant-rate linear error correcting code ECC with minimum relative distance δ, an efficient encoder Enc and an efficient decoder Dec recovering error up to $c_1 \cdot \delta$, where c_1 is a universal constant.*

We use a slight modification of the above construction, which is convenient when we want to guess-and-verify a circuit for the encoder.

▶ **Lemma 27.** *There is a constant $\delta > 0$ such that there is a constant-rate linear error correcting code ECC with minimum relative distance δ, an efficient encoder Enc and an efficient decoder Dec recovering error up to $c_1 \cdot \delta$, where c_1 is a universal constant. Moreover, each bit of the codeword depends on at most $n/2$ bits of the input.*

Proof. Given a message $x \in \{0,1\}^n$, we split it into three parts x_1, x_2, x_3, each of length between $\lfloor n/3 \rfloor$ and $\lceil n/3 \rceil$. Let Enc' and Dec' be the corresponding encoder and decoder of Lemma 26.

We construct our new error correcting code by setting $\mathsf{Enc}(x) := \mathsf{Enc}'(x_1) \circ \mathsf{Enc}'(x_2) \circ \mathsf{Enc}'(x_3)$. Given a codeword y, we split it into three strings y_1, y_2, y_3 of appropriate lengths, and let $\mathsf{Dec}(y) := \mathsf{Dec}'(y_1) \circ \mathsf{Dec}'(y_2) \circ \mathsf{Dec}'(y_3)$. ◀

2.5 Norms and Inequalities for Functions on Boolean Cube

For our lower bounds on approximate sums of functions, we will require a bit of Fourier analysis on Boolean functions. Here we introduce some notations and inequalities for real-valued functions on the Boolean hypercube. (See [36] for an excellent reference on this topic.)

Let $f : \{0,1\}^n \to \mathbb{R}$ be a function and $p \in \mathbb{R}^+$. We define

$$\|f\|_p := \left(\mathop{\mathbb{E}}_{x \sim \mathcal{U}_n} [|f(x)|^p] \right)^{1/p}.$$

We also define the infinity norm in the usual way:

$$\|f\|_\infty = \max_{x \in \{0,1\}^n} |f(x)|.$$

By the standard relations between different L_p-norms, for all $0 < p < q \leq \infty$, we have $\|f\|_p \leq \|f\|_q$.

For two functions $f, g : \{0,1\}^n \to \mathbb{R}$, we define their inner product as

$$\langle f, g \rangle := \mathop{\mathbb{E}}_{x \sim \mathcal{U}_n} [f(x) \cdot g(x)].$$

Note that the Cauchy-Schwarz inequality implies $\langle f, g \rangle \leq \|f\|_2 \cdot \|g\|_2$. We need the following simple lemma for this paper.

▶ **Lemma 28.** *For functions f_1, f_2 and g_1, g_2 from $\{0,1\}^n \to \mathbb{R}$ and positive $\varepsilon, \alpha \in \mathbb{R}$, suppose for all $i \in [2]$ we have:*

- *$\|f_i\|_2 \leq \alpha$ and $\|g_i\|_2 \leq \alpha$,*
- *$\|f_i - g_i\|_2 \leq \varepsilon$.*

Then $\langle f_1, f_2 \rangle - \langle g_1, g_2 \rangle \| \leq 2 \cdot \alpha \cdot \varepsilon$.

Proof. We have

$$\begin{aligned}\|\langle f_1, f_2\rangle - \langle g_1, g_2\rangle\| &\leq \|\langle f_1, f_2\rangle - \langle f_1, g_2\rangle\| + \|\langle f_1, g_2\rangle - \langle g_1, g_2\rangle\| \\ &\leq \|\langle f_1, f_2 - g_2\rangle\| + \|\langle f_1 - g_1, g_2\rangle\| \\ &\leq 2 \cdot \alpha \cdot \varepsilon.\end{aligned}$$

◀

2.6 Connections Between Nondeterministic Gap-UNSAT Algorithms and Circuit Lower Bounds

We also appeal to several known connections between Gap-UNSAT algorithms which improve upon exhaustive search and circuit lower bounds against nondeterministic time classes [49, 30, 41, 13].

▶ **Theorem 29** ([49]). *If Gap-UNSAT with gap $1 - 1/n^{10}$ for (general) circuits with n inputs and $\mathrm{poly}(n)$ size is solvable in $O(2^n/n^{\omega(1)})$ nondeterministic time, then NEXP doesn't have $\mathrm{poly}(n)$-size (general) circuits.*

▶ **Theorem 30** ([34]). *If there is an $\varepsilon > 0$ such that Gap-UNSAT with gap $1 - 1/n^{10}$ for (general) circuits with n inputs and $2^{\varepsilon n}$ size is solvable in $O(2^{n-\varepsilon n})$ nondeterministic time, then for every k there is a function in NP that does not have n^k-size (general) circuits.*

▶ **Theorem 31** (Corollary 12 in Tell [45], following [34]). *If there is a $\delta > 0$ and $c \geq 1$ such that Gap-UNSAT with gap $1 - 1/n^{10}$ for (general) circuits with n variables and m gates is solvable in $O(2^{n(1-\delta)} \cdot m^c)$ nondeterministic time, then for every unbounded $\alpha(n)$ such that $n^{\alpha(n)}$ is time-constructible, there is a function in $\mathsf{NTIME}[n^{\alpha(n)}]$ that is not in P/poly.*

▶ **Theorem 32** ([34]). *If there is an $\varepsilon > 0$ such that Gap-UNSAT with gap $1 - 1/n^{10}$ for (general) circuits with n inputs and 2^{n^ε} size is solvable in $O(2^{n-n^\varepsilon})$ nondeterministic time, then for every k there is a function in $\mathsf{NTIME}[n^{\mathrm{poly}(\log n)}]$ that does not have $n^{\log^k n}$-size (general) circuits.*

3 Equivalence Between Algorithmic Analysis of THR ∘ THR and of THR ∘ MAJ or MAJ ∘ MAJ

In this section, building on our new structure lemmas for THR∘THR circuits. We show several equivalence results between canonical circuit-analysis tasks (SAT or CAPP) of THR ∘ THR circuits and that of THR ∘ MAJ or MAJ ∘ MAJ circuits.

3.1 Poly-Size THR ∘ MAJ and THR ∘ THR are Equivalent for Circuit-Analysis Algorithms

We first show that, in terms of designing non-trivial circuit-analysis algorithms, THR ∘ THR and THR ∘ MAJ circuits are essentially *equivalent*.

▶ **Reminder of Theorem 1.** *The following two statements hold:*

- ***Equivalence of Non-Trivial SAT Algorithms**: There is a non-trivial SAT algorithm for THR ∘ MAJ circuits of $\mathrm{poly}(n)$-size* if and only if *there is such an algorithm for $\mathrm{poly}(n)$-size THR ∘ THR circuits.*
- ***Equivalence of Non-Trivial CAPP Algorithms With Constant Error**: For any constant $\delta > 0$, If there is a non-trivial CAPP algorithm with error δ for THR ∘ MAJ circuits of $\mathrm{poly}(n)$ size, then there is a non-trivial CAPP algorithm with error $\delta + 1/n$ for $\mathrm{poly}(n)$-size THR ∘ THR circuits.*

Proof. We begin with the first equivalence. We only have to show that a $2^n/n^{\omega(1)}$ time SAT algorithm for poly(n)-size THR $\circ$ MAJ circuits implies such an algorithm for THR $\circ$ THR circuits. By Lemma 16, given any THR $\circ$ THR circuit of poly(n) size, in poly(n) time we can construct an equivalent poly(n)-size Gap-OR $\circ$ THR $\circ$ MAJ circuit C. Applying the assumed SAT algorithm for THR $\circ$ MAJ circuits on all THR $\circ$ MAJ subcircuits of C completes the proof of the first equivalence.

For the second equivalence, given any THR $\circ$ THR circuit C of poly(n) size, we construct in poly(n) time a Gap-OR$_{1/n}$ $\circ$ THR $\circ$ MAJ circuit D that is equivalent to C, by Lemma 16. Let $D_1, D_2, \ldots, D_m$ be the THR $\circ$ MAJ subcircuits of C, where $m = \text{poly}(n)$.

By the definition of a Gap-OR$_{1/n}$ gate, for all $x \in \{0,1\}^n$, we have

$$\left| C(x) - \mathop{\mathbb{E}}_{i \in [m]} D_i(x) \right| \leq 1/n.$$

Therefore, to estimate $\mathbb{E}_{x \sim \mathcal{U}_n}[C(x)]$ within error $\delta + 1/n$, it suffices to estimate $\mathbb{E}_{x \sim \mathcal{U}_n}[D_i(x)]$ for each $i \in [m]$ within error δ. Applying the non-trivial CAPP algorithm for THR $\circ$ MAJ circuits from the assumption completes the proof. ◀

With an argument similar to the proof of Theorem 1 and using the fact that MAJ $\circ$ THR $\subseteq$ MAJ $\circ$ MAJ (potentially multiple times), it is not hard to generalize Theorem 1 to hold for TC circuits of any constant depth d.

▶ **Reminder of Corollary 2.** *The following two statements hold for any constant d:*

- ***Equivalence of Non-Trivial SAT Algorithms**: There is a non-trivial SAT algorithm for THR $\circ$ $\widehat{\text{LT}}_{d-1}$ circuits of* poly(n)*-size* if and only if *there is such an algorithm for* poly(n)*-size* LT_d *circuits.*
- ***Equivalence of Non-Trivial CAPP Algorithms With Constant Error**: For any constant $\varepsilon > 0$, if there is a non-trivial CAPP algorithm with error ε for THR $\circ$ $\widehat{\text{LT}}_{d-1}$ circuits of* poly(n)*-size, then there is a non-trivial CAPP algorithm with error $\varepsilon + 1/n$ for* poly(n)*-size* LT_d *circuits.*

3.2 Weaker Equivalence Between Poly-Size THR $\circ$ THR and MAJ $\circ$ MAJ

We also show a weaker equivalence for THR $\circ$ THR and MAJ $\circ$ MAJ circuits.

▶ **Reminder of Theorem 3.** *The following two statements hold:*

- ***Equivalence of $2^{(1-\varepsilon)n}$-time SAT Algorithms**: If SAT for* poly(n)*-size MAJ $\circ$ MAJ circuits is in $2^{(1-\varepsilon)n}$ time for some constant $\varepsilon > 0$, then SAT for* poly(n)*-size THR $\circ$ THR circuits is in $2^{(1-\varepsilon')n}$ time for some $\varepsilon' > 0$.*
- ***Equivalence of Non-Trivial CAPP Algorithms with Inverse Polynomial Error**: If there is a non-trivial CAPP algorithm with $1/poly(n)$ error for* poly(n)*-size MAJ $\circ$ MAJ circuits, then there is a non-trivial CAPP algorithm with* $1/\text{poly}(n)$ *error for* poly(n)*-size THR $\circ$ THR circuits.*

Proof. We begin with the first equivalence.

The first equivalence. Suppose we have a $2^{(1-\varepsilon_1)n}$ time SAT algorithm for poly(n) size MAJ $\circ$ MAJ circuits for a constant $\varepsilon_1 > 0$, and want to design a $2^{n-\Omega(n)}$ time SAT algorithm for poly(n) size THR $\circ$ THR circuits.

Let c be the hidden constant in the big-O of the fan-in of the top DOR gate from Lemma 17. Set $\varepsilon := \varepsilon_1/2c$, and apply Lemma 17 to the given poly(n)-size THR $\circ$ THR

circuit. We obtain an equivalent $\mathsf{DOR} \circ \mathsf{MAJ} \circ \mathsf{MAJ}$ circuit with top fan-in $2^{c\varepsilon n} = 2^{\varepsilon_1/2 \cdot n}$ and $\mathrm{poly}(n)$-size $\mathsf{MAJ} \circ \mathsf{MAJ}$ subcircuits. Then we can apply our SAT algorithm for $\mathrm{poly}(n)$-size $\mathsf{MAJ} \circ \mathsf{MAJ}$ circuits to solve the SAT problem for $\mathrm{poly}(n)$ size $\mathsf{THR} \circ \mathsf{THR}$ circuits, which completes the proof of the first equivalence.

The second equivalence. To show the second equivalence, suppose for all constants k', there is a CAPP algorithm for $\mathrm{poly}(n)$-size $\mathsf{MAJ} \circ \mathsf{MAJ}$ circuits with error $1/n^{k'}$. We have to design such an algorithm for $\mathrm{poly}(n)$-size $\mathsf{THR} \circ \mathsf{THR}$ circuits.

Given a $\mathsf{THR} \circ \mathsf{THR}$ circuit C of $s = \mathrm{poly}(n) \leq n^{c_1}$ size and a constant k, we want to estimate

$$\mathop{\mathbb{E}}_{x \sim \mathcal{U}_n}[C(x)] \tag{2}$$

within error $1/n^k$.

Since $\mathsf{THR} \subseteq \mathsf{DOR} \circ \mathsf{ETHR}$ (item (2) of Proposition 18), we can write C as a DOR of $m = \mathrm{poly}(s) = \mathrm{poly}(n)$ $\mathsf{ETHR} \circ \mathsf{THR}$ subcircuits $C_1, C_2, \ldots, C_m$. By the definition of DOR, we have

$$\mathop{\mathbb{E}}_{x \sim \mathcal{U}_n}[C(x)] = \mathop{\mathbb{E}}_{x \sim \mathcal{U}_n}\left[\sum_{i=1}^{m} C_i(x)\right] = \sum_{i=1}^{m} \mathop{\mathbb{E}}_{x \sim \mathcal{U}_n}[C_i(x)].$$

Therefore, in order to estimate (2) within error $1/n^k$, it suffices to estimate

$$\mathop{\mathbb{E}}_{x \sim \mathcal{U}_n}[C_i(x)]$$

within error $1/(m \cdot n^k)$ for each $i \in [m]$.

So fix an $i \in [m]$. Let $D = C_i$, and let D's top ETHR gate be G. By construction, G has weights of absolute value at most 2^{n^c}, for a constant c depending on c_1. Define $L : \{0,1\}^n \to \mathbb{Z}$ so that $L(x)$ is the value of the linear function associated with G on input x. That is, $D(x) = 1$ if and only if $L(x) = T$ for the threshold T of G.

Suppose we pick a random prime number p in the interval $[2, M]$, where $M = n^{2c} \cdot (2m \cdot n^k)^2 \leq \mathrm{poly}(n)$. Then for a fixed $x \in \{0,1\}^n$, if $L(x) \neq T$, the probability that $L(x) \equiv T \pmod{p}$ is less than $1/(2m \cdot n^k)$.

Recall that for a prime p and an ETHR gate $G(x) = [\sum_{i=1}^{n} w_i \cdot x_i = T]$, we use G^p to denote its "mod p" version (see Definition 41). Let D^p denote the circuit obtained by replacing the top G gate in D by G^p. For all $x \in \{0,1\}^n$, by the above discussion, we have

$$\left| D(x) - \mathop{\mathbb{E}}_{\text{prime } p \in [2,\, M]}[D^p(x)] \right| \leq 1/(2m \cdot n^k).$$

Therefore, in order to estimate $\mathbb{E}_{x \sim \mathcal{U}_n}[D(x)]$ within error $1/(m \cdot n^k)$, it suffices to estimate

$$\mathop{\mathbb{E}}_{x \sim \mathcal{U}_n}[D^p(x)]$$

for all primes $p \leq M$, within error $1/(2m \cdot n^k)$.

By Lemma 42, each D^p can be written as a DOR of $O(n)$ $\mathsf{EMAJ} \circ \mathsf{ETHR}$ circuits of $\mathrm{poly}(n)$ size. Since $\mathsf{EMAJ} \subseteq \mathsf{MAJ} \circ \mathsf{AND}_2$, $\mathsf{AND} \circ \mathsf{ETHR} \subseteq \mathsf{ETHR}$ and $\mathsf{MAJ} \circ \mathsf{ETHR} \subseteq \mathsf{MAJ} \circ \mathsf{MAJ}$ (items (6), (5), and (3) of Proposition 18), D^p can be further written as a DOR of cn $\mathsf{MAJ} \circ \mathsf{MAJ}$ circuits $D_1^p, D_2^p, \ldots, D_{cn}^p$ of $\mathrm{poly}(n)$ size, for a universal constant c.

Therefore, to estimate $\mathbb{E}_{x\sim\mathcal{U}_n}[D^p(x)]$ within error $1/(2m\cdot n^k)$, it suffices to estimate $\mathbb{E}_{x\sim\mathcal{U}_n}[D_i^p(x)]$ within error $1/(2m\cdot n^k\cdot cn)$, for each $i\in[cn]$.
Observe that $2m\cdot n^k\cdot cn\le \mathrm{poly}(n)$, and all D_i^p's are $\mathrm{poly}(n)$-size $\mathsf{MAJ}\circ\mathsf{MAJ}$ circuits. Applying the assumed CAPP algorithm completes the proof of the second equivalence. ◀

Again applying the fact that $\mathsf{MAJ}\circ\mathsf{THR}\subseteq\mathsf{MAJ}\circ\mathsf{MAJ}$, the generalization to TC circuits of any constant depth d is immediate.

▶ **Reminder of Corollary 4.** *The following two statements hold for any constant d:*

- ***Equivalence of $2^{(1-\varepsilon)n}$-time SAT Algorithms**: If SAT for $\widehat{\mathsf{LT}}_d$ circuits of $\mathrm{poly}(n)$-size is in $2^{(1-\varepsilon)n}$ time for a constant $\varepsilon>0$, then SAT for $\mathrm{poly}(n)$-size LT_d circuits is in $2^{(1-\varepsilon')n}$ time for some $\varepsilon'>0$.*
- ***Equivalence of Non-trivial CAPP Algorithms with inverse polynomial error**: If there is a non-trivial CAPP algorithm with $1/poly(n)$ error for $\mathrm{poly}(n)$-size $\widehat{\mathsf{LT}}_d$ circuits, then there is a non-trivial CAPP algorithm with $1/\mathrm{poly}(n)$ error for $\mathrm{poly}(n)$-size LT_d circuits.*

4 Tighter Connection Between Derandomization and Circuit Lower Bounds

In this section we show that $\mathcal{C}$ circuit lower bounds for NEXP or NP follow from better-than-2^n time derandomization of $\mathsf{AND}_3\circ\mathcal{C}$, $\mathsf{OR}_2\circ\mathcal{C}$, $\oplus_2\circ\mathcal{C}$ or $\mathsf{AND}_2\circ\mathcal{C}$ circuits.

▶ **Reminder of Theorem 5.** *There is an absolute constant $\delta>0$, such that for any typical circuit class $\mathcal{C}$, if one of the following holds:*

- *there is a non-trivial Gap-UNSAT algorithm with gap δ for $\mathrm{poly}(n)$-size $\mathsf{AND}_3\circ\mathcal{C}$ circuits, or*
- *there is a non-trivial CAPP algorithm with error δ for $\mathrm{poly}(n)$-size $\mathsf{OR}_2\circ\mathcal{C}$, $\oplus_2\circ\mathcal{C}$, or $\mathsf{AND}_2\circ\mathcal{C}$ circuits,*

then $\mathsf{NEXP}\not\subset\mathcal{C}$. Moreover, in the second bullet, $\mathcal{C}$ does not need to be closed under negation.

Proof. We use $\mathcal{U}_n$ to denote the uniform distribution on $\{0,1\}^n$.

We will show there is an absolute constant $\delta>0$, such that if one of the algorithmic assumptions of the theorem holds and $\mathsf{NEXP}\subset\mathcal{C}$, then $\mathsf{Gap\text{-}UNSAT}$ with gap $1-1/n^{10}$ for $\mathrm{poly}(n)$-size general circuits can be solved in $2^n/n^{\omega(1)}$ non-deterministic time. This proves the theorem, since by Theorem 29, we have $\mathsf{NEXP}\not\subset\mathsf{P}_{/\mathrm{poly}}$, which is a contradiction to $\mathsf{NEXP}\subset\mathcal{C}$.

We are given a $\mathrm{poly}(n)$-size general circuit $C:\{0,1\}^n\to\{0,1\}$ with the promise that either C is unsatisfiable, or C has at least $(1-1/n^{10})\cdot 2^n$ satisfying assignments. Our goal is to distinguish between these two cases in $2^n/n^{\omega(1)}$ non-deterministic time.

Let $\delta_1>0$ be the constant of Lemma 27. We fix a constant-rate linear error correcting code with minimum relative distance δ_1, as guaranteed by Lemma 27. Let $\mathsf{Enc}:\{0,1\}^n\to\{0,1\}^{cn}$ and $\mathsf{Dec}:\{0,1\}^{cn}\to\{0,1\}^n$ be the corresponding encoder and decoder, where $c\ge 1$ is a constant corresponding to the rate of the code. Let $\delta_{\mathsf{Dec}}=c_1\cdot\delta_1$, which is error rate that Dec can recover.

We also need a $\mathcal{C}$ circuit for the parity function on $n/2$ bits for computing Enc (by Lemma 27, the code is linear, and each output bit depends on at most $n/2$ input bits). By the assumption $\mathsf{NEXP}\subset\mathcal{C}$, the parity function must have a $\mathcal{C}$-circuit of $\mathrm{poly}(n)$ size. We can guess a $\mathcal{C}$-circuit $\mathsf{Par}_{n/2}$, and brute-force verify that it is correct in $2^{n/2}\cdot\mathrm{poly}(n)$ time.

Let $D : \{0,1\}^{cn} \to \{0,1\}$ be the circuit defined as $D(y) = \neg C(\mathsf{Dec}(y))$. Since C has $\mathrm{poly}(n)$ size and Dec is efficient, D also has $\mathrm{poly}(n)$ size. Then we can see

$$\Pr_{x\sim\mathcal{U}_n}[C(x)=0] = \Pr_{x\sim\mathcal{U}_n}[D(\mathsf{Enc}(x))=1] = \Pr_{x\sim\mathcal{U}_n}[(D,\mathsf{Enc}(x)) \in \mathsf{Circuit\text{-}Eval}].$$

With non-trivial Gap-UNSAT algorithms for poly-size $\mathsf{AND}_3 \circ \mathcal{C}$ circuits. We first prove the theorem under the first assumption. For that purpose we make use of a PCP of proximity system V for $\mathsf{Circuit\text{-}Eval}$, with $\delta_{\mathsf{PCPP}} < \delta_{\mathsf{Dec}}$, $r = O(\log n)$, $q = 3$ and a constant $s < 1$, whose existence is guaranteed by Lemma 24. We fix the circuit to be D, and write the verifier as $V(D)$.

We can view the verification of $V(D)$ as $m = 2^{r(|D|)} = \mathrm{poly}(n)$ many constraints on the oracle $y \circ \pi$. We can also assume $|\pi| = \ell = \mathrm{poly}(n)$. Suppose there are $F_1, F_2, \ldots, F_m$ constraints on $y \circ \pi$, each constraint is an OR on $q = 3$ variables or their negations.

Then the properties of PCP of proximity system translate to:

- If $y = \mathsf{Enc}(x)$ such that $C(x) = 0$, then $D(y) = 1$ and there is a proof $\pi \in \{0,1\}^\ell$ such that all constraints F_i's are satisfied by $y \circ \pi$.
- If $y = \mathsf{Enc}(x)$ such that $C(x) = 1$, then for all $z \in \{0,1\}^{cn}$ with $\mathsf{dist}(z,y) \le \delta_{\mathsf{Dec}}$, we have $D(z) = C(\mathsf{Dec}(z)) = C(x) = 0$. Therefore, y is δ_{Dec}-far from $\mathsf{Circuit\text{-}Eval}(D) = \{z : z \in \{0,1\}^{cn} \text{ and } (D,z) \in \mathsf{Circuit\text{-}Eval}\}$. Since $\delta_{\mathsf{Dec}} > \delta_{\mathsf{PCPP}}$, we have that for all proofs $\pi \in \{0,1\}^\ell$, at most a s fraction of constraints F_i's are satisfied by $y \circ \pi$.

When C is unsatisfiable, then there is a proof $\pi(x)$ for each $y = \mathsf{Enc}(x)$, such that $V(D)$ accepts $y \circ \pi(x)$ with probability 1. Note that by Lemma 24, such a proof $\pi(x)$ can be computed in polynomial time from y and D, which in particular means that $\pi(x)$ admits a polynomial-size circuit, hence each bit of $\pi(x)$ admits a $n_{\mathsf{proof}} = \mathrm{poly}(n)$ size $\mathcal{C}$ circuit (here we use the assumption that $\mathsf{NEXP} \subset \mathcal{C}$).

Next, we guess a list of n_{proof}-size $\mathcal{C}$ circuits $T_1, T_2, \ldots, T_\ell$ such that

$$T(x) = (T_1(x), T_2(x), \ldots, T_\ell(x))$$

is intended to be the proof $\pi(x)$ for $y = \mathsf{Enc}(x)$. Slightly abusing notation, we also use F_i to denote the function $F_i(x) := F_i(\mathsf{Enc}(x) \circ T(x))$. Since a bit of $\mathsf{Enc}(x)$ is just a parity on at most $n/2$ bits in x, and since $\mathcal{C}$ is typical, each F_i can be written as an $\mathsf{OR}_3 \circ \mathcal{C}$ circuit. We also set $E_i(x) = \neg F_i(x)$, which is an $\mathsf{AND}_3 \circ \mathcal{C}$ circuit.

Therefore, when C is unsatisfiable, by the previous discussion, on some guesses of the T_i's, we have

$$\Pr_{x\sim\mathcal{U}_n}[V(D)^{\mathsf{Enc}(x)\circ T(x)} = 1] = \mathop{\mathbb{E}}_{x\sim\mathcal{U}_n} \mathop{\mathbb{E}}_{i\in[m]}[F_i(x)] = 1.$$

Therefore, for all $i \in [m]$,

$$\mathop{\mathbb{E}}_{x\sim\mathcal{U}_n}[E_i(x)] = 0.$$

When C has at most $2^n/n^{10}$ unsatisfying assignments, for all possible $T_1, T_2, \ldots, T_\ell$, we have

$$\mathop{\mathbb{E}}_{x\sim\mathcal{U}_n} \mathop{\mathbb{E}}_{i\in[m]}[F_i(x)] \le 1/n^{10} + s.$$

By an averaging argument, there must be an i such that

$$\mathop{\mathbb{E}}_{x\sim\mathcal{U}_n}[F_i(x)] \le 1/n^{10} + s,$$

or equivalently

$$\mathop{\mathbb{E}}_{x\sim\mathcal{U}_n}[E_i(x)] \geq 1 - s - 1/n^{10} \geq \frac{1-s}{2}.$$

Next, we set $\delta = \frac{1-s}{2}$. When C is unsatisfiable, all E_i's are unsatisfiable on the correct guesses. When C has at most $1/n^{10} \cdot 2^n$ unsatisfying assignments, then for all guesses, there is at least one i such that E_i has at least $\delta \cdot 2^n$ satisfying assignments. Hence, solving Gap-UNSAT with gap δ for all E_i's suffices to non-deterministically distinguish between the two cases. By the first assumption, that takes $2^n/n^{\omega(1)}$ time, and the theorem follows from Theorem 29.

With non-trivial CAPP algorithms for $\mathrm{poly}(n)$-size $\mathsf{AND}_2 \circ \mathcal{C}$, $\oplus_2 \circ \mathcal{C}$ or $\mathsf{OR}_2 \circ \mathcal{C}$ circuits. The theorem under the second assumption can be proved similarly if we use the 2-query PCP of proximity system for Circuit-Eval instead, which is given by Lemma 25. The proof here is similar in parts to the one we just described for $\mathsf{AND}_3 \circ \mathcal{C}$; for completeness we will give the proof in full.

Now we make use of a PCP of proximity system V for Circuit-Eval, with $\delta_{\mathsf{PCPP}} < \delta_{\mathsf{Dec}}$, $r = O(\log n)$, $q = 2$ and constants $0 < s < c < 1$, whose existence is guaranteed by Lemma 25. We again fix the circuit to be D, and write the verifier as $V(D)$.

Similarly, we can view the verification of $V(D)$ as $m = 2^{r(|D|)} \leq \mathrm{poly}(n)$ many constraints on the oracle $y \circ \pi$. We can also assume $|\pi| = \ell \leq \mathrm{poly}(n)$. Suppose there are $F_1, F_2, \ldots, F_m$ constraints on $y \circ \pi$, where each constraint is a function on $q = 2$ coordinates of $y \circ \pi$.

Then the properties of PCP of proximity system translate to:

- If $y = \mathsf{Enc}(x)$ such that $C(x) = 0$, then $D(y) = 1$ and there is a proof $\pi \in \{0,1\}^\ell$ such that at least a c-fraction of F_i's are satisfied by $y \circ \pi$.
- If $y = \mathsf{Enc}(x)$ such that $C(x) = 1$, then for all $z \in \{0,1\}^{cn}$ with $\mathsf{dist}(z, y) \leq \delta_{\mathsf{Dec}}$, we have $D(z) = C(\mathsf{Dec}(z)) = C(x) = 0$. Therefore, y is δ_{Dec}-far from Circuit-Eval(D). Since $\delta_{\mathsf{Dec}} > \delta_{\mathsf{PCPP}}$, we have that for all proofs $\pi \in \{0,1\}^\ell$, at most an s-fraction of F_i's are satisfied by $y \circ \pi$.

If C is unsatisfiable, then there is a proof $\pi(x)$ for each $y = \mathsf{Enc}(x)$ that makes $V(D)$ accept $y \circ \pi(x)$ with probability at least c. By Lemma 24, such a proof $\pi(x)$ can be computed in polynomial time from y and D, which in particular means that $\pi(x)$ has a polynomial-size circuit. Therefore each output bit of $\pi(x)$ has an $n_{\mathsf{proof}} = \mathrm{poly}(n)$ size $\mathcal{C}$-circuit, from the assumption that $\mathsf{NEXP} \subset \mathcal{C}$.

The next step is to guess a list of n_{proof}-size $\mathcal{C}$ circuits $T_1, T_2, \ldots, T_\ell$ such that $T(x) = (T_1(x), T_2(x), \ldots, T_\ell(x))$ is supposed to the proof $\pi(x)$ given input $y = \mathsf{Enc}(x)$. Slightly abusing notation, F_i is also used to denote the function $F_i(x) := F_i(\mathsf{Enc}(x) \circ T(x))$.

When C is unsatisfiable, by the previous discussion, there is a guess of T_i's such that

$$\Pr_{x\sim\mathcal{U}_n}[V(D)^{\mathsf{Enc}(x)\circ T(x)} = 1] = \mathop{\mathbb{E}}_{x\sim\mathcal{U}_n}\mathop{\mathbb{E}}_{i\in[m]}[F_i(x)] \geq c.$$

When C has at most $2^n/n^{10}$ unsatisfying assignments, then for all possible $T_1, T_2, \ldots, T_\ell$, we have

$$\mathop{\mathbb{E}}_{x\sim\mathcal{U}_n}\mathop{\mathbb{E}}_{i\in[m]}[F_i(x)] \leq 1/n^{10} + s.$$

Now set $\delta_1 := \frac{c-s}{2}$. In order for us to non-deterministically distinguish between the above two cases, it suffices to estimate

$$\mathop{\mathbb{E}}_{x\sim\mathcal{U}_n}[F_i(x)]$$

to within δ_1, for each $i \in [m]$.

Since each output bit of $\mathsf{Enc}(x)$ is simply a parity on at most $n/2$ bits of x, each F_i can be written as a function $F_i(x) = P(C_1(x), C_2(x))$, where C_1, C_2 are two $\mathcal{C}$ circuits, and P is a function from $\{0,1\}^2 \to \{0,1\}$. (Recall that in this case, we do not require $\mathcal{C}$ to be closed under negation.)

Now we write P as a polynomial:

$$P(z_1, z_2) = \sum_{S \subseteq [2]} \alpha_S \cdot \prod_{i \in S} z_i = \sum_{S \subseteq [2]} \alpha_S \cdot \bigwedge_{i \in S} z_i,$$

where each coefficient $\alpha_S \in [-4, 4]$. Given two $\mathcal{C}$ circuits C_1, C_2, to estimate

$$\mathop{\mathbb{E}}_{x \sim \mathcal{U}_n}[P(C_1(x), C_2(x))] = \sum_{S \subseteq [2]} \alpha_S \cdot \mathop{\mathbb{E}}_{x \sim \mathcal{U}_n}\left[\bigwedge_{i \in S} C_i(x)\right]$$

within error δ_1, it suffices to estimate each

$$\mathop{\mathbb{E}}_{x \sim \mathcal{U}_n}\left[\bigwedge_{i \in S} C_i(x)\right]$$

within error $\delta = \delta_1/16$. Finally, we can apply our assumed non-trivial CAPP algorithm for $\mathrm{poly}(n)$-size $\mathsf{AND}_2 \circ \mathcal{C}$ circuits to non-deterministically distinguish the two cases, and the theorem follows from Theorem 29.

When we only have non-trivial CAPP algorithms for $\oplus_2 \circ \mathcal{C}$ or $\mathsf{OR}_2 \circ \mathcal{C}$ circuits, we can simply write P in the basis of OR functions or $\oplus$ functions instead. That is, we can write

$$P(z_1, z_2) = \sum_{S \subseteq [2]} \alpha'_S \cdot \bigoplus_{i \in S} z_i,$$

or

$$P(z_1, z_2) = \sum_{S \subseteq [2]} \alpha''_S \cdot \bigvee_{i \in S} z_i.$$

The rest of the argument is the same as the case of $\mathsf{AND}_2 \circ \mathcal{C}$ circuits. ◀

Using Theorem 30, the following theorem can be proved with the same argument as of Theorem 5.

▶ **Reminder of Theorem 6.** *There is an absolute constant $\alpha > 0$, such that for any typical circuit class $\mathcal{C}$, if there is a constant δ such that one of the following holds:*

- *Gap-UNSAT for $2^{\delta n}$-size $\mathsf{AND}_3 \circ \mathcal{C}$ circuits with gap α can be solved in $2^{n-\delta n}$ time, or*
- *CAPP for $2^{\delta n}$-size $\mathsf{OR}_2 \circ \mathcal{C}$, $\oplus_2 \circ \mathcal{C}$, or $\mathsf{AND}_2 \circ \mathcal{C}$ circuits with error α can be solved in $2^{n-\delta n}$ time,*

then for every k there is a function in NP that doesn't have n^k-size $\mathcal{C}$ circuits. Moreover, in the second bullet, $\mathcal{C}$ does not need to be closed under negation.

5 Approaches For THR ∘ THR Circuit Lower Bounds

In this section we propose approaches for proving $\mathsf{NEXP} \not\subset \mathsf{THR} \circ \mathsf{THR}$. We will see that surprisingly weak algorithms suffice for proving this lower bound.

Applying Theorem 5 and the fact that $\oplus_2 \circ \mathsf{THR} \circ \mathsf{THR} \subseteq \mathsf{THR} \circ \mathsf{THR}$, we first show that $\mathsf{NEXP} \not\subset \mathsf{THR} \circ \mathsf{THR}$ would follow from a non-trivial CAPP algorithm for $\mathrm{poly}(n)$-size $\mathsf{THR} \circ \mathsf{THR}$ circuits.

▶ **Reminder of Theorem 8.** *There is an absolute constant $\delta > 0$, such that if δ-error CAPP for $\mathrm{poly}(n)$-size THR∘THR circuits can be solved in $2^n/n^{\omega(1)}$ time, then NEXP $\not\subset$ THR∘THR. The same is true with SAT in place of CAPP.*

Proof. The theorem for CAPP follows directly from the fact that $\oplus_2 \circ \mathsf{THR} \circ \mathsf{THR} \subseteq \mathsf{THR} \circ \mathsf{THR}$ (item (7) of Proposition 18) and Theorem 5.

Suppose SAT for $\mathrm{poly}(n)$-size $\mathsf{THR} \circ \mathsf{THR}$ circuits can be solved in $2^n/n^{\omega(1)}$ time. By Theorem 5, it suffices to give a $2^n/n^{\omega(1)}$ time algorithm for solving SAT for $\mathsf{AND}_3 \circ \mathsf{THR} \circ \mathsf{THR}$ circuits of $\mathrm{poly}(n)$ size (note that Gap-UNSAT is easier than SAT).

Given such an $\mathsf{AND}_3 \circ \mathsf{THR} \circ \mathsf{THR}$ circuit C, we first use the fact that $\mathsf{THR} \subseteq \mathsf{DOR} \circ \mathsf{ETHR}$ (item (2) of Proposition 18) to transform it into a $\mathrm{poly}(n)$ size $\mathsf{AND}_3 \circ \mathsf{DOR} \circ \mathsf{ETHR} \circ \mathsf{ETHR}$ circuit C'.

Treating the DOR as an addition gate (that has at most one true input), and the AND_3 as a multiplication, we can apply distributivity to the circuit. Together with the fact that $\mathsf{AND} \circ \mathsf{ETHR} \subseteq \mathsf{ETHR}$ (item (5) of Proposition 18), C' is then equivalent to a $\mathsf{DOR} \circ \mathsf{ETHR} \circ \mathsf{ETHR}$ circuit C'' of $\mathrm{poly}(n)$ size.

Finally, observe that solving SAT for C'' can be reduced to solving SAT for its $\mathrm{poly}(n)$ $\mathsf{ETHR} \circ \mathsf{ETHR}$ subcircuits, and note that $\mathsf{ETHR} \circ \mathsf{ETHR}$ can be converted efficiently into $\mathsf{THR} \circ \mathsf{THR}$ (item (4) of Proposition 18). Therefore, applying the $2^n/n^{\omega(1)}$ time SAT algorithm for $\mathrm{poly}(n)$-size $\mathsf{THR} \circ \mathsf{THR}$ circuits from the assumption completes the proof. ◀

In fact, similar results apply to TC circuits of any constant depth d (i.e., LT_d circuits). The following theorem can be proved in exactly the same way.

▶ **Reminder of Theorem 9.** *There is an absolute constant $\delta > 0$, such that for any constant d, if CAPP for $\mathrm{poly}(n)$-size LT_d circuits with error δ can be solved in $2^n/n^{\omega(1)}$ time, then NEXP $\not\subset \mathsf{LT}_d$. The same is true with SAT in place of CAPP.*

Now, combing Theorem 8 and our equivalence theorems (Theorem 1 and Theorem 3), the following corollary follows immediately.

▶ **Reminder of Corollary 10.** *There is an absolute constant $\delta > 0$, such that if one of the following holds:*

1. *CAPP (or SAT) for $\mathrm{poly}(n)$-size THR∘MAJ circuits with error δ can be solved in $2^n/n^{\omega(1)}$ time, or*
2. *CAPP for $\mathrm{poly}(n)$-size MAJ∘MAJ circuits with $1/poly(n)$ error can be solved in $2^n/n^{\omega(1)}$ time.*

Then NEXP $\not\subset$ THR∘THR.

6 Lower Bounds for $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathcal{C}$ Circuits

We now present our lower bounds for various $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathcal{C}$ circuits. In the following we slightly abuse notation, by also using $\mathcal{C}$ to denote a class of functions from $\{0,1\}^n \to \mathbb{R}$. Note that Boolean circuit classes are special cases of real-valued function classes. We also assume $\mathcal{C}$ contains the constant functions $\mathbf{0}$ and $\mathbf{1}$ for simplicity.

We first define the Sum-Product problem over functions from $\mathcal{C}$.

Sum-Product over $\mathcal{C}$

Given k functions $f_1, \ldots, f_k : \{0,1\}^n \to \mathbb{R}$ from $\mathcal{C}$, compute

$$\sum_{x \in \{0,1\}^n} \prod_{i=1}^{k} f_i(x).$$

6.1 The Main Challenge: Guessed $\mathsf{Sum}_\varepsilon \circ \mathcal{C}$ Circuits Could be Invalid

The main idea is to follow the proof of Theorem 5. Suppose we are given $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathcal{C}$ circuits C_1 and C_2 computing two Boolean functions f_1 and f_2, respectively. One can see that their product $C(x) := C_1(x) \cdot C_2(x)$ (which is a real function on $\{0,1\}^n$), is an $(3 \cdot \varepsilon)$-approximation to $f_1 \wedge f_2$, for small enough $\varepsilon > 0$.

Therefore, if we simply computed

$$\sum_{x \in \{0,1\}^n} C_1(x) \cdot C_2(x), \tag{3}$$

we would have estimated $\Pr_{x \in \{0,1\}^n}[(f_1(x) \wedge f_2(x)) = 1]$ within 3ε. Note that if C_1 (C_2) is a linear sum of m_1 (m_2) $\mathcal{C}$-circuits, then the above quantity can be reduced to $m_1 \cdot m_2$ instances of the sum-product problem over $\mathcal{C}$.

However, the above reasoning does not complete the proof. The problem is that in the proof of Theorem 5, one actually has to *guess* the $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathcal{C}$ circuits which are supposed to compute the PCPP proofs. It could well be the case that our guessed representations are *not valid at all*. That is, it could be that for some $x \in \{0,1\}^n$, $\sum_{i=1}^S \alpha_i \cdot C_i(x)$ is much larger than 1, or much less than 0. If C_1 and C_2 are not valid $\mathsf{Sum}_\varepsilon \circ \mathcal{C}$ circuits to begin with, then the quantity (3) would not be useful.

6.2 Testing Whether a Linear Representation is Close to Boolean

This issue also occurs in Williams' lower bounds on $\mathsf{Sum} \circ \mathcal{C}$ circuits [48]. There, the problem is solved by using a clever algorithm to verify whether a given $\mathsf{Sum} \circ \mathcal{C}$ circuit is valid. In particular, the test of whether a $\mathsf{Sum} \circ \mathcal{C}$ outputs 0 or 1 on every Boolean input is effectively reduced to a small number of Sum-Product calls. But this argument crucially uses the fact that the $\mathsf{Sum} \circ \mathcal{C}$ must output one of two discrete values on every Boolean input. It appears to be much harder to verify that a given $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathcal{C}$ circuit is valid.

We will later show that it suffices to test whether a given $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathcal{C}$ circuit is close to a Boolean function *with respect to ℓ_2 distance*, in which case we know how to get an algorithm.

It will be convenient to introduce some notation. Let $d_{\mathsf{bin}}(z) = \min_{b \in \{0,1\}} |z - b|$. Intuitively, $d_{\mathsf{bin}}(z)$ measures how close z is to a bit-value. For a function $f : \{0,1\}^n \to \mathbb{R}$, define its closest binary function bin_f as follows: for all $x \in \{0,1\}$, if $f(x) \geq 1/2$, $\mathsf{bin}_f(x) := 1$, otherwise $\mathsf{bin}_f(x) := 0$. By definition, for any $p > 0$ we have

$$\|f - \mathsf{bin}_f\|_p = \left(\mathop{\mathbb{E}}_{x \sim \mathcal{U}_n} [|d_{\mathsf{bin}}(f(x))|^p] \right)^{1/p},$$

and

$$\|f - \mathsf{bin}_f\|_\infty = \max_{x \in \{0,1\}^n} |d_{\mathsf{bin}}(f(x))|.$$

Let $f = \sum_{i=1}^S \alpha_i \cdot C_i$ be a linear combination of functions from $\mathcal{C}$; we wish to verify that f is a $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathcal{C}$ circuit for some Boolean function. With respect to the above definitions, f is a valid $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathcal{C}$ circuit for some Boolean function if and only if $\|f - \mathsf{bin}_f\|_\infty \leq \varepsilon$.

The following algorithm shows that, given an algorithm for evaluating the Sum-Product of 4 functions from $\mathcal{C}$, the algorithm can be used to distinguish between the case that $\|f - \mathsf{bin}_f\|_\infty$ is small and the case that $\|f - \mathsf{bin}\|_2$ is large.

▶ **Lemma 33.** *For $S \in \mathbb{N}$, suppose we are given S reals $\{\alpha_i\}_{i\in[S]}$, S functions from $\mathcal{C}$ $\{C_i\}_{i\in[S]}$, and parameter $\varepsilon < 0.01$. Suppose Sum-Product of 4 functions on n bits from $\mathcal{C}$ can be solved in $T(n)$ time. Let $f = \sum_{i=1}^{S} \alpha_i \cdot C_i$.*

There is an algorithm A such that:

- *If $\|f - \mathsf{bin}_f\|_\infty \le \varepsilon$, then A always accepts. (That is, if $\sum_{i=1}^{S} \alpha_i \cdot C_i$ is a valid $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathcal{C}$ circuit for* some *Boolean function, then A always accepts.)*
- *If $\|f - \mathsf{bin}_f\|_2 \ge 3 \cdot \varepsilon$, then A always rejects.*
- *Otherwise, A can output anything.*
- *A runs in $T(n) \cdot (S+1)^4 + 2^{o(n)}$ time.*

Proof. We define a polynomial of degree 4,

$$P(z) := z^2 \cdot (1-z)^2,$$

to approximate $d_{\mathsf{bin}}(z)$. Simple calculations confirm the following facts about $P(z)$:

- $P(z) \le d_{\mathsf{bin}}(z)^2 \cdot (1 + d_{\mathsf{bin}}(z))^2$, and
- $P(z) \ge d_{\mathsf{bin}}(z)^2 \cdot 2^{-2}$.

When $d_{\mathsf{bin}}(z) \le \varepsilon$, we have $P(z) \le \varepsilon^2 \cdot (1+\varepsilon)^2$. This means that if $\|f - \mathsf{bin}_f\|_\infty \le \varepsilon$, then we have

$$\mathop{\mathbb{E}}_{x\sim\mathcal{U}_n}[P(f(x))] \le \varepsilon^2 \cdot (1+\varepsilon)^2 \le \varepsilon^2 \cdot (1+0.01)^2.$$

On the other hand, if $\|f - \mathsf{bin}_f\|_2 \ge 3 \cdot \varepsilon$, then by definition we have

$$\mathop{\mathbb{E}}_{x\sim\mathcal{U}_n}\left[d_{\mathsf{bin}}(f(x))^2\right] \ge (3\cdot\varepsilon)^2,$$

therefore

$$\mathop{\mathbb{E}}_{x\sim\mathcal{U}_n}[P(f(x))] \ge (3/2)^2 \cdot \varepsilon^2.$$

Therefore, it suffices to compute

$$\mathop{\mathbb{E}}_{x\sim\mathcal{U}_n}[P(f(x))] \tag{4}$$

to distinguish between these two cases.

Expanding out $P(f(x)) = P(\sum_{i=1}^{S} \alpha_i \cdot C_i)$, it can be written as a $\mathbb{R}$-sum of at most $(S+1)^4$ products of 4 functions from $\mathcal{C}$. By rearranging the order of summation (summing all $(S+1)^4$ terms first), we see that (4) can be evaluated by making at most $(S+1)^4$ calls to the assumed Sum-Product algorithm. Assuming that algorithm runs in $T(n)$ time, the sum (4) can be evaluated in time $T(n) \cdot (S+1)^4 + 2^{o(n)}$. ◀

6.3 Meta-Theorem for $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathcal{C}$ Lower Bounds

Now we are ready to prove the following meta theorem for lower bounds on $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathcal{C}$ circuits.

▶ **Theorem 34.** *Suppose every $C \in \mathcal{C}$ has a $\mathrm{poly}(n)$-bit representation, where each C can be evaluated in $\mathrm{poly}(n)$ time. Assume there is a $\delta > 0$ such that for all constant integers $k > 0$, there is a $\mathrm{poly}(n) \cdot 2^{n-\delta n}$-time algorithm for computing the Sum-Product of k functions on n bits from $\mathcal{C}$. Then:*

- *For every k and constant $\varepsilon < 1/2$, there is a function in NP without $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathcal{C}$ circuits of n^k sparsity.*
- *For every unbounded function $\alpha(n)$ such that $n^{\alpha(n)}$ is time-constructible, NTIME$[n^{\alpha(n)}]$ doesn't have $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathcal{C}$ circuits of polynomial sparsity for all constant $\varepsilon < 1/2$.*

The most important component in the proof of the above meta-theorem is an argument that we can solve Gap-UNSAT faster, given a non-trivial algorithm for evaluating Sum-Product of functions from $\mathcal{C}$ and the assumption that Circuit-Eval has a small $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathcal{C}$ circuit. Formally, we have the following lemma, whose proof follows similar reasoning as the proof of Theorem 5, while taking care of the issue that a guessed $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathcal{C}$ circuit may not be a valid one with the algorithm from Lemma 33.

▶ **Lemma 35.** *There is an absolute constant $\varepsilon > 0$ such that if:*

- *there is a $\delta > 0$ such that for all integers $k \leq 4$, there is a $\mathrm{poly}(n) \cdot 2^{n-\delta n}$-time algorithm for computing the Sum-Product of k functions on n bits from $\mathcal{C}$, and*
- *Circuit-Eval has a $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathcal{C}$ circuit of sparsity n^k for some $k > 0$,*

then there is a non-deterministic $2^{n-\delta n} \cdot \mathrm{poly}(n, s)$ time algorithm for Gap-UNSAT with gap $1 - 1/n^{10}$, on general (fan-in 2) circuits with n inputs and s gates.

Proof. Suppose we are given an s-size general circuit $C : \{0,1\}^n \to \{0,1\}$ with the promise that either C is unsatisfiable or C has at least $(1 - 1/n^{10}) \cdot 2^n$ satisfying assignments. We want to distinguish between these two cases in $2^{n-\delta n} \cdot \mathrm{poly}(n, s)$ non-deterministic time.

Let δ_1 be the constant from Lemma 27. Fix a constant-rate linear error correcting code with minimum relative distance δ_1, as guaranteed by Lemma 27, letting $\mathsf{Enc} : \{0,1\}^n \to \{0,1\}^{cn}$ and $\mathsf{Dec} : \{0,1\}^{cn} \to \{0,1\}^n$ be the corresponding encoder and decoder, for a constant c corresponding to the rate of the code. Let $\delta_{\mathsf{Dec}} = c_1 \cdot \delta_1$, which is error rate that Dec can recover.

We also need a $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathcal{C}$ circuit for the parity function on $n/2$ bits for computing Enc (by Lemma 27, the code is linear and each output bit depends on at most $n/2$ input bits). Applying the second assumption of the theorem, and the fact that parity reduces easily to Circuit-Eval (parity has linear-size circuits), there is a $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathcal{C}$ circuit of sparsity $n_{\mathsf{parity}} = n^{O(k)}$ for the parity function on $n/2$ inputs. We can guess such a $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathcal{C}$ circuit $\mathsf{Par}_{n/2}$ of n_{parity} size, and verify it is correct in $2^{n/2} \cdot \mathrm{poly}(n_{\mathsf{parity}}) = 2^{n/2} \cdot \mathrm{poly}(n)$ time, as in the proof of Theorem 5.

Let $D : \{0,1\}^{cn} \to \{0,1\}$ be the circuit defined as $D(y) = \neg C(\mathsf{Dec}(y))$. Since C is of s size and Dec is efficient, D is of size $n_D = \mathrm{poly}(n, s)$.

Then we observe that

$$\Pr_{x \sim \mathcal{U}_n}[C(x) = 0] = \Pr_{x \sim \mathcal{U}_n}[D(\mathsf{Enc}(x)) = 1] = \Pr_{x \sim \mathcal{U}_n}[(D, \mathsf{Enc}(x)) \in \mathsf{Circuit\text{-}Eval}].$$

Now we make use of a PCP of proximity system V for Circuit-Eval, with parameters $\delta_{\mathsf{PCPP}} < \delta_{\mathsf{Dec}}$, $r = O(\log n)$, $q = 2$, and constants $0 < s < c < 1$, with existence guaranteed by Lemma 25. We fix the circuit to be D, and write the verifier as $V(D)$.

We can view the verification of $V(D)$ as $m = 2^{r(|D|)} = \mathrm{poly}(n_D) \leq \mathrm{poly}(n, s)$ constraints on the oracle $y \circ \pi$. We can also assume $|\pi| = \ell \leq \mathrm{poly}(n, s)$. Suppose there are m constraints $F_1, F_2, \ldots, F_m$ on $y \circ \pi$, where each constraint is a function on two coordinates of $y \circ \pi$. Then the properties of the PCP of proximity yield the following consequences:

- If $y = \mathsf{Enc}(x)$ such that $C(x) = 0$, then $D(y) = 1$ and there is a proof $\pi \in \{0,1\}^\ell$ such that such that at least a c-fraction of the F_i's are satisfied by $y \circ \pi$.

- If $y = \mathsf{Enc}(x)$ such that $C(x) = 1$, then for all $z \in \{0,1\}^{cn}$ with $\mathsf{dist}(z, y) \leq \delta_{\mathsf{Dec}}$, we have $D(z) = C(\mathsf{Dec}(z)) = C(x) = 0$. Therefore, y is δ_{Dec}-far from $\mathsf{Circuit\text{-}Eval}(D)$. Since $\delta_{\mathsf{Dec}} > \delta_{\mathsf{PCPP}}$, we have that for all proofs $\pi \in \{0,1\}^{\ell}$, at most an s-fraction of the F_i's are satisfied by $y \circ \pi$.

When C is unsatisfiable, it means there is a proof $\pi(x)$ for each $y = \mathsf{Enc}(x)$, so that $V(D)$ accepts $y \circ \pi(x)$ with probability at least c. Note that by Lemma 25, such a proof $\pi(x)$ can be computed in polynomial time from y and D, which in particular means that $\pi(x)$ admits a $\mathrm{poly}(n, n_D) = \mathrm{poly}(n, s)$-size circuit. By our second assumption, each bit of $\pi(x)$ therefore has a $\widetilde{\mathsf{Sum}}_{\varepsilon} \circ \mathcal{C}$ circuit of sparsity $n_{\mathsf{proof}} = \mathrm{poly}(n, s)^{O(k)} \leq \mathrm{poly}(n, s)$.

Now, we guess a list of (presumably) $\widetilde{\mathsf{Sum}}_{\varepsilon} \circ \mathcal{C}$ circuits $T_1, T_2, \ldots, T_{\ell}$ each of sparsity n_{proof}, and denote $H_i = \mathsf{bin}_{T_i}$. We want $H(x) = (H_1(x), H_2(x), \ldots, H_{\ell}(x))$ to be the proof $\pi(x)$ for the input $y = \mathsf{Enc}(x)$. Let each $T_i = \sum_{j=1}^{n_{\mathsf{proof}}} \alpha_{i,j} \cdot E_{i,j}$, where each $\alpha_{i,j} \in \mathbb{R}$ and $E_{i,j} \in \mathcal{C}$. Slightly abusing notation, we also use F_i to denote the function $F_i(x) := F_i(\mathsf{Enc}(x) \circ H(x))$.

First, for all i, we apply the test of Lemma 33 on T_i with parameter ε. We reject immediately if some test rejects. Note that by Lemma 33 and our first assumption, all the tests take $2^{n-\delta n} \cdot \mathrm{poly}(n, s)$ time in total.

If all guesses are valid $\widetilde{\mathsf{Sum}}_{\varepsilon} \circ \mathcal{C}$ circuits, (that is, $\|H_i - T_i\|_{\infty} \leq \varepsilon$ for all i), then all the tests are passed by Lemma 33. Furthermore if all tests passed, we know that $\|H_i - T_i\|_2 \leq 3 \cdot \varepsilon$ for all $i \in [\ell]$ by Lemma 33.

Therefore, when C is unsatisfiable, by the previous discussion, there is some guess of T_i's such that

$$\Pr_{x \sim \mathcal{U}_n}[V(D)^{\mathsf{Enc}(x) \circ H(x)} = 1] = \Pr_{x \sim \mathcal{U}_n} \Pr_{i \in [m]}[F_i(x) = 1] \geq c.$$

When C has at most $2^n/n^{10}$ unsatisfying assignments, then for all possible $T_1, T_2, \ldots, T_{\ell}$, we have

$$\Pr_{x \sim \mathcal{U}_n} \Pr_{i \in [m]}[F_i(x)] \leq 1/n^{10} + s.$$

Note that F_i is a function on two coordinates of $\mathsf{Enc}(x) \circ H(x)$. In particular, since each bit of $\mathsf{Enc}(x)$ is a just a parity of some inputs in x (it is a linear code), we only need to estimate the following quantity for a function $P : \{0,1\}^2 \to \{0,1\}$:

$$\mathop{\mathbb{E}}_{x \sim \mathcal{U}_n}[P(L_1(x), L_2(x))], \tag{5}$$

where for each function $L_i(x)$, we have an approximate linear representation $T_i = \sum_{j=1}^{n_{\mathsf{final}}} \alpha_{i,j} \cdot E_{i,j}$, such that $\|T_i - L_i\|_2 \leq 3 \cdot \varepsilon$, where each $E_{i,j} \in \mathcal{C}$ and $n_{\mathsf{final}} = \max(n_{\mathsf{proof}}, n_{\mathsf{parity}}) \leq \mathrm{poly}(n, s)$.[13] (Note that when $L_i(x)$ is a bit in the error correcting code, we can simply use the guessed circuit $\mathsf{Par}_{n/2}$.)

Let $\varepsilon_2 = \frac{c-s}{2}$. In order to non-deterministically distinguish between the above two cases, we only have to estimate (5) within error ε_2.

We can write $P : \{0,1\}^2 \to \{0,1\}$ as a multi-linear polynomial, with

$$P(z) := \sum_{S \subseteq [2]} \alpha_S \cdot \prod_{i \in S} z_i,$$

[13] Here we don't need the fact that F_i is an OR on variables or their negations.

where each $\alpha_S \in [-2^2, 2^2]$. Therefore, to estimate (5) within ε_2, we only need to estimate

$$\mathop{\mathbb{E}}_{x\sim\mathcal{U}_n}\left[\prod_{i\in S} L_i(x)\right]$$

within error $\varepsilon_2/16$, for each $|S| \geq 1$. (when $S = \emptyset$, it is 1 by definition.)

Now, instead of the above, we compute

$$\mathop{\mathbb{E}}_{x\sim\mathcal{U}_n}\left[\prod_{i\in S} T_i(x)\right]. \tag{6}$$

When $|S| = 1$ and in particular $S = \{i\}$, we have

$$\mathop{\mathbb{E}}_{x\sim\mathcal{U}_n}[L_i(x)] - \mathop{\mathbb{E}}_{x\sim\mathcal{U}_n}[T_i(x)] \leq \|L_i - T_i\|_1 \leq \|L_i - T_i\|_2 \leq 3\cdot\varepsilon.$$

When $|S| = 2$, we want to bound

$$|\langle L_1, L_2\rangle - \langle T_1, T_2\rangle|.$$

Since L_i is Boolean, we have $\|L_i\|_2 = 1$, and therefore $\|T_i\|_2 \leq 1 + 3\cdot\varepsilon$ by the triangle inequality. By Lemma 28, we have

$$|\langle L_1, L_2\rangle - \langle T_1, T_2\rangle| \leq (1+3\varepsilon)\cdot 2\cdot 3\varepsilon.$$

Now we set ε such that $(1+3\varepsilon)\cdot 2\cdot 3\varepsilon = \varepsilon_2/16$.

Finally, for each $S \subseteq [2]$, computing (6) can be reduced to $n_{\mathsf{final}}^{|S|} \leq \mathrm{poly}(n,s)$ evaluations of Sum-Products of $|S| \leq 2$ functions on n bits from $\mathcal{C}$. By assumption, thse evaluations can be computed in $2^{n-\delta n}\cdot\mathrm{poly}(n,s)$ time, which completes the proof. ◀

Now we are ready to prove Theorem 34.

Proof of Theorem 34. For the first consequence, assume every function in NP has a $\widetilde{\mathsf{Sum}}_\varepsilon$ circuit of n^k sparsity, for some fixed $k > 0$ and $0 < \varepsilon < 0.5$. Let ε_1 be the absolute constant specified in Lemma 35. By Lemma 19, there is a polynomial P of degree $d = O(1)$, such that for all $b \in \{0,1\}$, if $|z - b| \leq \varepsilon$ then $|P(z) - b| \leq \varepsilon_1$.

Let $L : \{0,1\}^\star \to \{0,1\}$ be any function in NP and $\sum_{i=1}^{n^k} \alpha_i \cdot C_i$ be the $\widetilde{\mathsf{Sum}}_\varepsilon \circ \mathcal{C}$ circuit for $L_n : \{0,1\}^n \to \{0,1\}$, where each $C_i \in \mathcal{C}$.

Consider the function

$$P\left(\sum_{i=1}^{n^k} \alpha_i \cdot C_i\right). \tag{7}$$

By the definition of P, for all $x \in \{0,1\}^n$, we have

$$\left|P\left(\sum_{i=1}^{n^k} \alpha_i \cdot C_i(x)\right) - L_n(x)\right| \leq \varepsilon_1.$$

Expanding the expression of (7) into a sum of products, we obtain a $\widetilde{\mathsf{Sum}}_{\varepsilon_1} \circ \mathcal{C}^{\otimes d}$ circuit for L_n of sparsity $n^{k'}$, where $\mathcal{C}^{\otimes d}$ consists of all possible products of d functions from $\mathcal{C}$, and $k' \leq O(d\cdot k) \leq O(k)$. Observe that the Sum-Product problem for at most 4 functions from

$\mathcal{C}^{\otimes d}$ is simply the Sum-Product problem for at most $4 \cdot d$ functions from $\mathcal{C}$, which admits a $\text{poly}(n) \cdot 2^{n-\delta n}$ time algorithm by assumption.

Since Circuit-Eval $\in$ NP, both conditions of Lemma 35 are now satisfied for $\mathcal{C}^{\otimes d}$. Therefore Gap-UNSAT with gap $1 - 1/n^{10}$ for s-gate n-input circuits of fan-in 2 has a $2^{n-\delta n} \cdot \text{poly}(n, s)$ time non-deterministic algorithm. It follows from Lemma 30 that, for every k, there is a function in NP which does not have general circuits of n^k size and fan-in 2. This is a contradiction, since every $\widetilde{\text{Sum}}_\varepsilon \circ \mathcal{C}$ circuit of n^k sparsity can be simulated by an $n^{\alpha \cdot k}$-size general fan-in-2 circuit, for a universal constant $\alpha \geq 1$.

The second consequence follows the same way, by applying Theorem 31 instead. ◀

6.4 Lower Bounds for $\widetilde{\text{Sum}}_\varepsilon \circ$ THR/ReLU/$\mathbb{F}_p$-polynomials

In order to apply Theorem 34, we need the following algorithms from [48], for computing the Sum-Product of $O(1)$ functions from the function classes we care about.

▶ **Lemma 36** (Theorem 4.1 in [48]). *The Sum-Product of k THR functions on n variables (with weight in $[-n^n, n^n]$) can be computed in $2^{n/2} \cdot n^{O(k)}$ time.*

▶ **Lemma 37** (Theorem 5.1 in [48]). *The Sum-Product of k ReLU functions on n variables (with weight in $[-W, W]$) can be computed in $2^{n/2} \cdot n^{O(k)} \cdot \text{poly}(k, n, \log W)$ time.*

▶ **Lemma 38** (Theorem 6.1 in [48]). *The Sum-Product of k degree-d polynomials $p_1, \ldots, p_k \in \mathbb{F}_p[x_1, \ldots, x_n]$ can be computed in $p^{2k} \cdot (1.9^n + 2^{n-n/(6dp)}) \cdot \text{poly}(n)$ time.*

Applying Theorem 34 with the above algorithms for computing the Sum-Product for functions from THR, ReLU and $O(1)$-degree $\mathbb{F}_p$-polynomials, Theorem 12, Theorem 13 and Theorem 14 follow immediately.

6.5 Lower Bounds for $\widetilde{\text{Sum}}_\varepsilon \circ \text{ACC}^0 \circ$ THR

Theorem 15 follows via a similar argument as Theorem 34, and the known #SAT algorithms for $\text{ACC}^0 \circ$ THR [52]. Formally, we prove

▶ **Reminder of Theorem 15.** *For every $d, m \geq 1$ and $\varepsilon \in [0, 0.5)$, there is a $b \geq 1$ and an $f \in \text{NTIME}[n^{\log^b n}]$ that does not have $\widetilde{\text{Sum}}_\varepsilon \circ \text{AC}^0_d[m] \circ \text{THR}$ circuits of n^a size, for every a.*

Using the argument of Lemma 35, we can show:

▶ **Lemma 39.** *There is an absolute constant $\varepsilon > 0$ such that if the following two conditions hold:*

- *there is a $\delta > 0$ such that for all integer $k \leq 4$, there is a $\text{poly}(n) \cdot 2^{n-n^\delta}$-time algorithm for computing the Sum-Product of k functions on n bits from $\mathcal{C}$, and*
- *Circuit-Eval has a $\widetilde{\text{Sum}}_\varepsilon \circ \mathcal{C}$ of sparsity n^k for some $k > 0$.*

Then there is non-deterministic $2^{n-n^\delta} \cdot \text{poly}(n, s)$ time algorithm for Gap-UNSAT with gap $1 - 1/n^{10}$ and a general fan-in-2 circuit with n input and s gates.

Theorem 15 then follows from exactly the same arguments as that of Theorem 34, combining the following two facts:

1. For every depth d and integer $m \geq 2$, there is an $\varepsilon > 0$ such that the Sum-Product of $O(1)$ $\text{AC}^0_d[m] \circ$ THR circuits of 2^{n^ε} size can be computed in 2^{n-n^ε} time. This simply applies the algorithm for counting satisfying assignments of $\text{AC}^0_d[m] \circ$ THR circuits ([51]).

2. If for some $\alpha > 0$ there is a nondeterministic $2^{n-n^{\alpha}}$-time Gap-UNSAT algorithm with gap $1 - 1/n^{10}$ for $2^{n^{\alpha}}$-size circuits, then for every $a \geq 1$, there is a $b \geq 1$ such that $\mathsf{NTIME}[n^{\log^b n}]$ does not have $n^{\log^a n}$-size circuits (this is a theorem of Murray and Williams [34]).

6.6 A Note on the Coefficients in the $\widetilde{\mathsf{Sum}}_{\varepsilon} \circ \mathcal{C}$ Circuits

In our proof, we have to *guess* a $\widetilde{\mathsf{Sum}}_{\varepsilon} \circ \mathcal{C}$ circuit, so it is crucial that all $\widetilde{\mathsf{Sum}}_{\varepsilon} \circ \mathcal{C}$ circuits we consider have "reasonable" coefficients, with in $\mathrm{poly}(n)$-bit complexity. When all functions in $\mathcal{C}$ are Boolean-valued, the following proposition provides this guarantee.

▶ **Proposition 40.** *Let $\varepsilon \in [0, 0.5)$ be a constant of bit complexity b.*[14] *Let $\mathcal{C}$ be a class of functions with co-domain $\{0,1\}$, and let C be a $\mathsf{Sum}_{\varepsilon} \circ \mathcal{C}$ circuit of sparsity s for a Boolean function $f : \{0,1\}^n \to \{0,1\}$. There is an equivalent $\widetilde{\mathsf{Sum}}_{\varepsilon} \circ \mathcal{C}$ circuit C' such that every weight in the linear combination of $\mathcal{C}$ has the form j/k, where both j and k are integers in $[-s^{\mathrm{poly}(s,b)}, s^{\mathrm{poly}(s,b)}]$.*

Proof. Let C be a linear combination of s functions from $\mathcal{C}$. We may assume without loss of generality that these s functions are linearly independent. The problem of finding coefficients for these s functions to ε-approximate a given boolean function f is equivalent to finding a solution to a certain linear programming instance $\|Ax - b\|_{\infty} \leq \varepsilon$ in s unknowns over the rationals, where $b \in \{0,1\}^{2^n}$ represents the truth-table of the function f and $A \in \{0,1\}^{2^n \times s}$.

Standard results from the theory of linear programming show that, if the instance is feasible, then there is a valid solution corresponding to the unique solution of a linear system where some of the inequalities are tight (that is $(Ax - b)_i = \varepsilon$ or $(Ax - b)_i = -\varepsilon$). Then proposition then follows from Cramer's rule. ◀

The case for $\widetilde{\mathsf{Sum}}_{\varepsilon} \circ \mathsf{ReLU}$ circuits is more involved. Luckily, Maass [32] showed that the weights for such a circuit of sparsity s needs only $\mathrm{poly}(s, n)$ bits of precision.

7 Structure Lemmas for THR ∘ THR Circuits

In this section we present our structure lemmas for $\mathsf{THR} \circ \mathsf{THR}$ circuits. We first need a simple construction, which will be used in both proofs.

▶ **Definition 41** (Mod p Exact Threshold Gate)**.** *Let G be an ETHR gate with n inputs, p be a prime and G^p be the "mod p" version of G. That is, let L and T be the corresponding linear function and threshold of G, $G^p(x) := [L(x) \equiv T \pmod p]$.*

▶ **Lemma 42.** *Let G be an ETHR gate with n inputs and p be a prime. Then G^p can be written as a DOR ∘ ETHR circuit such that*

- *The top DOR gate has $O(n)$ fan-in.*
- *All ETHR gates have positive weights and thresholds smaller than $O(np)$.*[15]

Proof. Let $w_1, w_2, \ldots, w_n$ and T be the corresponding weights and threshold of G. Reduce each weight w_i in G to $w_i \bmod p$ (the corresponding integer between 0 and $p-1$). This yields another circuit with associate top linear function $L'(x)$, whose value is always at most np. Setting $t = T \bmod p$, $L(x) \equiv T \pmod p$ is equivalent to $L'(x) = t + k \cdot p$ for

[14] That is, we assume ε can be specified as the ratio of two b-bit integers.
[15] Therefore, when $p \leq \mathrm{poly}(n)$, the ETHR gate can be seen as an EMAJ gate.

some $k \in \{0, 1, 2, \ldots, n\}$. Therefore, by taking an OR over all possible k on the condition $L'(x) = t + k \cdot p$, it is a disjoint OR, and we obtain the equivalent DOR $\circ$ ETHR circuit. ◀

7.1 Proof of Structure Lemma I

We begin with the proof of Structure Lemma I for THR $\circ$ THR circuits (restated below).

▶ **Reminder of Lemma 16.** *Let n be number of inputs, $s = s(n) \geq n$ be a size parameter and $\delta = \delta(n)$ be the error parameter. Every s-size THR $\circ$ THR circuit C is equivalent to a Gap-OR$_\delta$ $\circ$ THR $\circ$ MAJ circuit such that:*

- *The top Gap-OR$_\delta$ gate has* $\text{poly}(s, \delta^{-1})$ *fan-in.*
- *Each sub THR $\circ$ MAJ circuit has size* $\text{poly}(s, \delta^{-1})$.

Moreover, the reduction can be computed in deterministic $\text{poly}(s, \delta^{-1})$ *time.*

Proof. Let C' be the given THR $\circ$ THR circuit. By negating some of its input gates (THR is closed under negation), we may assume all weights in the top THR gate of C' are ≤ 0. Since every THR can be converted into a DOR $\circ$ ETHR (item (2) of Proposition 18), C' can be transformed into an equivalent THR $\circ$ ETHR circuit C of size $t = \text{poly}(s)$.

Let $G_1, G_2, \ldots, G_t$, $w_1, w_2, \ldots, w_t$ be the ETHR gates on the bottom layer and their corresponding weights in the top gate of C. By assumption, we also have $w_i \leq 0$ for all i. Let T be the threshold of the top gate. For all inputs x of n bits, we have

$$C(x) = \left[\sum_{i=1}^{t} w_i \cdot G_i(x) \geq T \right].$$

By construction, we may assume that the weights in G_i are bounded by 2^{n^c} for a constant c. Suppose we fix an input x, and let p be a random prime from 2 to $n^{2c} \cdot t^2 \cdot \delta^{-1} = \text{poly}(s, \delta^{-1})$. With probability at least $1 - \delta/t$, we have $G_i^p(x) = G_i(x)$. Let C^p be the circuit obtained by replacing all G_i's in C by corresponding G_i^p's.

When $C(x) = 1$, it follows from a union bound that $C^p(x) = C(x) = 1$ with probability at least $1 - \delta$. When $C(x) = 0$, note that for all primes p, we have $G_i^p(x) \geq G_i(x)$ for all i, therefore we must have $\sum_i^s w_i \cdot G_i^p(x) \leq \sum_i^s w_i \cdot G_i(x) < T$ (all w_i's are ≤ 0) and $C^p(x) = 0$.

Therefore C is equivalent to a Gap-OR$_\delta$ over *all* C^p's, for every prime p (recall their total number is $\text{poly}(s, \delta^{-1})$). By Lemma 42, each C^p can be expressed as a $\text{poly}(s, \delta^{-1})$-size THR $\circ$ EMAJ circuit. Converting each THR $\circ$ EMAJ into a THR $\circ$ MAJ (item (8) of Proposition 18) completes the proof. ◀

7.2 Proof of Structure Lemma II

Now we turn to proving Lemma 17. The proof has two steps, provided by Lemma 43 and Lemma 45.

▶ **Lemma 43** (Weight Reduction at the Top THR gate)**.** *Every size-s THR$_d$ $\circ$ $\mathcal{C}$ circuit (having a top THR gate of fan-in d) is equivalent to a DOR $\circ$ ETHR $\circ$ $\mathcal{C}$ circuit such that:*

- *The top DOR gate has* $\text{poly}(d)$ *fan-in.*
- *Each ETHR gate has fan-in d, with positive weights and threshold value, all of which are less than* $\text{poly}(d) \cdot 2^n$.
- *The $\mathcal{C}$-part is unchanged.*

The same statement also holds for ETHR$_d$ $\circ$ $\mathcal{C}$ circuits. Moreover, the reductions can be computed in randomized $\text{poly}(s)$ *time.*

Proof. We only consider the $\mathsf{THR}_d \circ \mathcal{C}$ case (the $\mathsf{ETHR}_d \circ \mathcal{C}$ case is even easier).

Let C be the given circuit. First, by the fact that $\mathsf{THR} \subseteq \mathsf{DOR} \circ \mathsf{ETHR}$ (item (2) of Proposition 18), C can be transformed to an equivalent $\mathsf{DOR} \circ \mathsf{ETHR} \circ \mathcal{C}$ circuit C'.

Let G be a ETHR gate in C'; note that G has fan-in d. Let D be the subcircuit with top gate G. By construction, G has weights of absolute value at most $M_{\mathsf{old}} = 2^{\mathrm{poly}(d)}$.

Next, we define $L : \{0,1\}^n \to \mathbb{Z}$ such that $L(x)$ is the value of the linear function associated with the gate G when the input is x. That is $D(x) = 1$ if and only if $L(x) = T$ for the threshold T of G.

Pick a random prime number m in the interval $[2, M_{\mathsf{new}}]$, where $M_{\mathsf{new}} = d^c \cdot 2^n$ and c is a sufficiently large constant. For a fixed $x \in \{0,1\}^n$, if $L(x) \neq T$, the probability that $L(x) \equiv T \pmod m$ is smaller than

$$\frac{\log(M_{\mathsf{old}})}{M_{\mathsf{new}}/\ln(M_{\mathsf{new}})} = \frac{\mathrm{poly}(d)}{\Theta(2^n \cdot d^c/(n + c\log d))} \leq d^{-c/2}/2^n,$$

for a sufficiently large c. Applying the union bound over all inputs x, with probability at least $1 - d^{-c/2}$, we have $L(x) \equiv T \pmod m$ if and only if $L(x) = T$ for all $x \in \{0,1\}^n$.

Finally, applying Lemma 42 with prime m, we can replace G with an equivalent $\mathsf{DOR} \circ \mathsf{ETHR}$ subcircuit, whose ETHR gates have positive weights and thresholds smaller than $\mathrm{poly}(d) \cdot 2^n$.

Union-bounding over all ETHR gates, and choosing c to be a large enough constant, this completes the randomized reduction. ◀

▶ Remark 44. One can observe that the above reduction indeed only introduces *one-sided error*. That is, even if it chooses some "bad" primes, the resulting circuit D satisfies the property that $D(x) = 1$ whenever $C(x) = 1$.

▶ **Lemma 45** (Decomposition of the top ETHR gate). *Given an $\mathsf{ETHR}_d \circ \mathcal{C}$ circuit C (a circuit with a top ETHR gate of fan-in d) of size s and a real $\varepsilon \in \left(\frac{\log d}{n}, 1\right)$, suppose the top ETHR gate in C has positive weights and threshold smaller than 2^{2n}. C is equivalent to a $\mathsf{DOR} \circ \mathsf{MAJ} \circ \mathsf{AND}_2 \circ \mathcal{C}$ circuit such that:*

- *The top DOR gate has $2^{O(\varepsilon n)}$ fan-in.*
- *Each MAJ gate has fan-in $d^{O(1/\varepsilon)}$.*
- *The $\mathcal{C}$ part is unchanged.*

Moreover, the reduction can be computed in deterministic

$$2^{O(\varepsilon n)} \cdot d^{O(1/\varepsilon)} + \mathrm{poly}(s)$$

time.

Proof. Let G_{top} be the top ETHR in C, and let $G_1, G_2, \ldots, G_d$ be its input gates. Let w_i's and T be the weights and the threshold of G_{top} and $L(x)$ be the associated linear function. We have for all $x \in \{0,1\}^n$ that

$$L(x) = \sum_{i=1}^{d} w_i \cdot G_i(x).$$

Observe that the binary representations of w_i's and T are of length at most $\log(2^{2n}) \leq 2n$. Break each of their binary representations into $D = \left\lceil \frac{\varepsilon \cdot n}{\log d} \right\rceil$ blocks, where each block has $B \leq 2/\varepsilon \cdot \log d$ bits. Let $w_{i,j}, T_j \in [2^B - 1]$ be the values of w_i's and T's j-th block, respectively (where blocks are numbered from the least significant bit to the most significant bit).

Consider adding the $w_i \cdot G_i(x)$'s in base 2^B, keeping track of all $D-1$ carries on each position, except for the highest one. Let $c = (c_1, c_2, \ldots, c_{D-1}) \in \{0, 1, \ldots, d-1\}^{D-1}$ be such a carry sequence. Observe that $\sum_{i=1}^{d} w_i \cdot G_i(x) = T$ with carry sequence c if and only if for all $j \in [D]$:

$$\sum_{i=1}^{d} w_{i,j} \cdot G_i(x) + c_{j-1} = T_j + 2^B \cdot c_j,$$

where we set C_D and C_0 to be 0 for notational convenience. That is, after we fix the carries c_j's for all j, the sums $\sum_{i=1}^{d} w_{i,j} \cdot G_i(x)$ are also forced to be $T_j^c = T_j + 2^B \cdot c_j - c_{j-1}$. Therefore, consider the sum

$$\sum_{j=1}^{\varepsilon \cdot n} \left(\sum_{i=1}^{d} w_{i,j} \cdot G_i(x) - T_j^c \right)^2.$$

Checking whether this sum is at most 0 can be formulated as a $\mathrm{poly}(d) \cdot 2^{O(B)} = d^{O(1/\varepsilon)}$ size $\mathsf{MAJ} \circ \mathsf{AND}_2$ subcircuit, with input gates $G_1, G_2, \ldots, G_d$.

Each of these addition checks corresponds to one carry sequence. By enumerating all possible d^{D-1} carry sequences, the above transforms G_{top} into a $\mathsf{DOR} \circ \mathsf{MAJ} \circ \mathsf{AND}_2$ subcircuit with input gates $G_1, G_2, \ldots, G_d$, having top fan-in:

$$d^{D-1} = d^{O(\varepsilon \cdot n / \log d)} = 2^{O(\varepsilon \cdot n)},$$

which completes the proof. ◀

Finally, the Structure Lemma II for $\mathsf{THR} \circ \mathsf{THR}$ circuits follows from applying Lemma 43 and Lemma 45 in the appropriate way.

▶ **Reminder of Lemma 17.** *Let n be the number of inputs and let $s = s(n) \leq 2^{o(n)}$ be a size parameter. Let $\varepsilon \in \left(\frac{\log s}{n}, 1\right)$. Every s-size $\mathsf{THR} \circ \mathsf{THR}$ circuit C is equivalent to a $\mathsf{DOR} \circ \mathsf{MAJ} \circ \mathsf{MAJ}$ circuit such that:*

- *The top DOR gate has $2^{O(\varepsilon n)}$ fan-in.*
- *Each sub $\mathsf{MAJ} \circ \mathsf{MAJ}$ circuit has size $s^{O(1/\varepsilon)}$.*

The reduction can be computed in randomized $2^{O(\varepsilon n)} \cdot s^{O(1/\varepsilon)}$ *time.*

Proof. First, since $\mathsf{THR} \subseteq \mathsf{DOR} \circ \mathsf{ETHR}$ (item (2) of Proposition 18), C is equivalent to a $\mathrm{poly}(s)$-size $\mathsf{THR} \circ \mathsf{ETHR}$ circuit C_1. Moreover, we can convert C into C_1 in polynomial time.

Second, we apply Lemma 43 to transform C_1 into a $\mathsf{DOR}_{\mathrm{poly}(s)} \circ \mathsf{ETHR} \circ \mathsf{ETHR}$ circuit C_2, such that all middle-layer ETHR gates have positive weights and thresholds smaller than $\mathrm{poly}(s) \cdot 2^n < 2^{2n}$.

Third, we apply Lemma 45 to C_2, which changes all middle-layer ETHR gates of C_2 into $\mathsf{DOR} \circ \mathsf{MAJ} \circ \mathsf{AND}_2$ subcircuits, with top gate fan-in $2^{O(\varepsilon \cdot n)}$. This yields a $\mathsf{DOR} \circ \mathsf{MAJ} \circ \mathsf{AND} \circ \mathsf{ETHR}$ circuit. Converting the remaining $\mathsf{AND} \circ \mathsf{ETHR}$ subcircuits into ETHR's (item (5) of Proposition 18), we obtain a $\mathsf{DOR}_{2^{O(\varepsilon \cdot n)}} \circ \mathsf{MAJ} \circ \mathsf{ETHR}$ circuit where all $\mathsf{MAJ} \circ \mathsf{ETHR}$ subcircuits have size at most $s^{O(1/\varepsilon)}$.

Finally, converting each $\mathsf{MAJ} \circ \mathsf{ETHR}$ into a $\mathsf{MAJ} \circ \mathsf{MAJ}$ (item (3) of Proposition 18) completes the reduction. The running time bound follows from plugging in the time bounds of Lemma 43 and Lemma 45. ◀

Setting the parameter ε carefully in Lemma 17, we have the following corollary.

▶ **Corollary 46.** *Let n be the number of inputs and let $s = s(n) \leq 2^{o(n)}$ be a size parameter. Let $\varepsilon \in \left(\frac{\log s}{n}, 1\right)$. Every s-size $\mathsf{THR} \circ \mathsf{THR}$ circuit C is equivalent to a $\mathsf{DOR} \circ \mathsf{MAJ} \circ \mathsf{MAJ}$ circuit C' such that:*

- *The top DOR gate of C' has $s^{O(1/\varepsilon)}$ fan-in.*
- *Every sub $\mathsf{MAJ} \circ \mathsf{MAJ}$ circuit of C' has size $2^{O(\varepsilon \cdot n)}$.*

The reduction can be computed in randomized $2^{O(\varepsilon n)} \cdot s^{O(1/\varepsilon)}$ time.

References

1. Amir Abboud and Karl Bringmann. Tighter Connections Between Formula-SAT and Shaving Logs. In *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, pages 8:1–8:18, 2018. doi:10.4230/LIPIcs.ICALP.2018.8.
2. Eric Allender and Michal Koucký. Amplifying lower bounds by means of self-reducibility. *J. ACM*, 57(3):14:1–14:36, 2010. doi:10.1145/1706591.1706594.
3. Josh Alman, Timothy M. Chan, and R. Ryan Williams. Polynomial Representations of Threshold Functions and Algorithmic Applications. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 467–476, 2016. doi:10.1109/FOCS.2016.57.
4. Kazuyuki Amano and Akira Maruoka. On the Complexity of Depth-2 Circuits with Threshold Gates. In *Mathematical Foundations of Computer Science 2005, 30th International Symposium, MFCS 2005, Gdansk, Poland, August 29 - September 2, 2005, Proceedings*, pages 107–118, 2005. doi:10.1007/11549345_11.
5. Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof Verification and the Hardness of Approximation Problems. *J. ACM*, 45(3):501–555, 1998. doi:10.1145/278298.278306.
6. Sanjeev Arora and Shmuel Safra. Probabilistic Checking of Proofs: A New Characterization of NP. *J. ACM*, 45(1):70–122, 1998. doi:10.1145/273865.273901.
7. László Babai, Kristoffer Arnsfelt Hansen, Vladimir V Podolskii, and Xiaoming Sun. Weights of exact threshold functions. In *International Symposium on Mathematical Foundations of Computer Science*, pages 66–77. Springer, 2010.
8. Swapnam Bajpai, Vaibhav Krishan, Deepanshu Kush, Nutan Limaye, and Srikanth Srinivasan. A #SAT Algorithm for Small Constant-Depth Circuits with PTF Gates. In *10th Innovations in Theoretical Computer Science Conference, ITCS 2019, January 10-12, 2019, San Diego, California, USA*, pages 8:1–8:20, 2019. doi:10.4230/LIPIcs.ITCS.2019.8.
9. Paul Beame, Stephen A. Cook, and H. James Hoover. Log Depth Circuits for Division and Related Problems. *SIAM J. Comput.*, 15(4):994–1003, 1986. doi:10.1137/0215070.
10. Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil P. Vadhan. Short PCPs Verifiable in Polylogarithmic Time. In *20th Annual IEEE Conference on Computational Complexity (CCC 2005), 11-15 June 2005, San Jose, CA, USA*, pages 120–134, 2005. doi:10.1109/CCC.2005.27.
11. 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. doi:10.1137/S0097539705446810.
12. Eli Ben-Sasson and Emanuele Viola. Short PCPs with Projection Queries. In *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part I*, pages 163–173, 2014. doi:10.1007/978-3-662-43948-7_14.
13. Eli Ben-Sasson and Emanuele Viola. Short PCPs with projection queries. In *International Colloquium on Automata, Languages, and Programming*, pages 163–173. Springer, 2014.

14 Mark Bun, Robin Kothari, and Justin Thaler. The polynomial method strikes back: tight quantum query bounds via dual polynomials. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 297–310, 2018. doi:10.1145/3188745.3188784.

15 Mark Bun and Justin Thaler. A Nearly Optimal Lower Bound on the Approximate Degree of AC^0. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 1–12, 2017. doi:10.1109/FOCS.2017.10.

16 Arkadev Chattopadhyay and Nikhil S. Mande. A Short List of Equalities Induces Large Sign Rank. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 47–58, 2018. doi:10.1109/FOCS.2018.00014.

17 Ruiwen Chen and Rahul Santhanam. Improved Algorithms for Sparse MAX-SAT and MAX-k-CSP. In *Theory and Applications of Satisfiability Testing - SAT 2015 - 18th International Conference, Austin, TX, USA, September 24-27, 2015, Proceedings*, pages 33–45, 2015. doi:10.1007/978-3-319-24318-4_4.

18 Irit Dinur. The PCP theorem by gap amplification. *J. ACM*, 54(3):12, 2007. doi:10.1145/1236457.1236459.

19 Jürgen Forster, Matthias Krause, Satyanarayana V. Lokam, Rustam Mubarakzjanov, Niels Schmitt, and Hans Ulrich Simon. Relations Between Communication Complexity, Linear Arrangements, and Computational Complexity. In *FST TCS 2001: Foundations of Software Technology and Theoretical Computer Science, 21st Conference, Bangalore, India, December 13-15, 2001, Proceedings*, pages 171–182, 2001. doi:10.1007/3-540-45294-X_15.

20 M. R. Garey, David S. Johnson, and Larry J. Stockmeyer. Some Simplified NP-Complete Graph Problems. *Theor. Comput. Sci.*, 1(3):237–267, 1976. doi:10.1016/0304-3975(76)90059-1.

21 Mikael Goldmann, Johan Håstad, and Alexander A. Razborov. Majority Gates VS. General Weighted Threshold Gates. *Computational Complexity*, 2:277–300, 1992. doi:10.1007/BF01200426.

22 Hans Dietmar Groeger and György Turán. A linear lower bound for the size of threshold circuits. *Bulletin-European Association For Theoretical Computer Science*, 50:220–220, 1993.

23 András Hajnal, Wolfgang Maass, Pavel Pudlák, Mario Szegedy, and György Turán. Threshold Circuits of Bounded Depth. *J. Comput. Syst. Sci.*, 46(2):129–154, 1993. doi:10.1016/0022-0000(93)90001-D.

24 Kristoffer Arnsfelt Hansen and Vladimir V. Podolskii. Exact Threshold Circuits. In *Proceedings of the 25th Annual IEEE Conference on Computational Complexity, CCC 2010, Cambridge, Massachusetts, June 9-12, 2010*, pages 270–279, 2010. doi:10.1109/CCC.2010.33.

25 Kristoffer Arnsfelt Hansen and Vladimir V. Podolskii. Polynomial threshold functions and Boolean threshold circuits. *Inf. Comput.*, 240:56–73, 2015. doi:10.1016/j.ic.2014.09.008.

26 William Hesse, Eric Allender, and David A. Mix Barrington. Uniform constant-depth threshold circuits for division and iterated multiplication. *J. Comput. Syst. Sci.*, 65(4):695–716, 2002. doi:10.1016/S0022-0000(02)00025-9.

27 Thomas Hofmeister. A Note on the Simulation of Exponential Threshold Weights. In *Computing and Combinatorics, Second Annual International Conference, COCOON '96, Hong Kong, June 17-19, 1996, Proceedings*, pages 136–141, 1996. doi:10.1007/3-540-61332-3_146.

28 Russell Impagliazzo, Ramamohan Paturi, and Michael E. Saks. Size-Depth Tradeoffs for Threshold Circuits. *SIAM J. Comput.*, 26(3):693–707, 1997. doi:10.1137/S0097539792282965.

29 Russell Impagliazzo, Ramamohan Paturi, and Stefan Schneider. A Satisfiability Algorithm for Sparse Depth Two Threshold Circuits. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 479–488, 2013. doi:10.1109/FOCS.2013.58.

30 Hamid Jahanjou, Eric Miles, and Emanuele Viola. Local Reductions. In *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part I*, pages 749–760, 2015. doi:10.1007/978-3-662-47672-7_61.

31 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 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 633–643, 2016. doi:10.1145/2897518.2897636.

32 Wolfgang Maass. Bounds for the Computational Power and Learning Complexity of Analog Neural Nets. *SIAM J. Comput.*, 26(3):708–732, 1997. doi:10.1137/S0097539793256041.

33 Saburo Muroga, Iwao Toda, and Satoru Takasu. Theory of majority decision elements. *Journal of the Franklin Institute*, 271(5):376–418, 1961.

34 Cody Murray and R. 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, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 890–901, 2018. doi:10.1145/3188745.3188910.

35 Noam Nisan. The communication complexity of threshold gates. *Combinatorics, Paul Erdos is Eighty*, 1:301–315, 1993.

36 Ryan O'Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2014. URL: http://www.cambridge.org/de/academic/subjects/computer-science/algorithmics-complexity-computer-algebra-and-computational-g/analysis-boolean-functions.

37 Ramamohan Paturi and Michael E. Saks. Approximating Threshold Circuits by Rational Functions. *Inf. Comput.*, 112(2):257–272, 1994. doi:10.1006/inco.1994.1059.

38 John H. Reif and Stephen R. Tate. On Threshold Circuits and Polynomial Computation. *SIAM J. Comput.*, 21(5):896–908, 1992. doi:10.1137/0221053.

39 Theodore J Rivlin. *An introduction to the approximation of functions*. Courier Corporation, 2003.

40 Vwani P. Roychowdhury, Alon Orlitsky, and Kai-Yeung Siu. Lower bounds on threshold and related circuits via communication complexity. *IEEE Trans. Information Theory*, 40(2):467–474, 1994. doi:10.1109/18.312169.

41 Rahul Santhanam and Ryan Williams. On Medium-Uniformity and Circuit Lower Bounds. In *Proceedings of the 28th Conference on Computational Complexity, CCC 2013, K.lo Alto, California, USA, 5-7 June, 2013*, pages 15–23, 2013. doi:10.1109/CCC.2013.40.

42 Alexander A. Sherstov. Algorithmic polynomials. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 311–324, 2018. doi:10.1145/3188745.3188958.

43 Daniel A. Spielman. Linear-time encodable and decodable error-correcting codes. *IEEE Trans. Information Theory*, 42(6):1723–1731, 1996. doi:10.1109/18.556668.

44 Suguru Tamaki. A Satisfiability Algorithm for Depth Two Circuits with a Sub-Quadratic Number of Symmetric and Threshold Gates. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:100, 2016. URL: http://eccc.hpi-web.de/report/2016/100.

45 Roei Tell. Proving that prBPP=prP is as hard as "almost" proving that P $\neq$ NP. *Electronic Colloquium on Computational Complexity (ECCC)*, 25:3, 2018. URL: https://eccc.weizmann.ac.il/report/2018/003.

46 Roei Tell. Quantified derandomization of linear threshold circuits. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 855–865, 2018. doi:10.1145/3188745.3188822.

47 R. Ryan Williams. Natural Proofs versus Derandomization. *SIAM J. Comput.*, 45(2):497–529, 2016. doi:10.1137/130938219.

48 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, June 22-24, 2018, San Diego, CA, USA*, pages 6:1–6:24, 2018. doi:10.4230/LIPIcs.CCC.2018.6.

49 Ryan Williams. Improving Exhaustive Search Implies Superpolynomial Lower Bounds. *SIAM J. Comput.*, 42(3):1218–1244, 2013. doi:10.1137/10080703X.

50 Ryan Williams. Towards NEXP versus BPP? In *Computer Science - Theory and Applications - 8th International Computer Science Symposium in Russia, CSR 2013, Ekaterinburg, Russia, June 25-29, 2013. Proceedings*, pages 174–182, 2013. doi:10.1007/978-3-642-38536-0_15.

51 Ryan Williams. New algorithms and lower bounds for circuits with linear threshold gates. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 194–202, 2014. doi:10.1145/2591796.2591858.

52 Ryan Williams. Nonuniform ACC circuit lower bounds. *Journal of the ACM (JACM)*, 61(1):2, 2014.

A Constructions of Super Query-Efficient PCP of Proximity Systems

In this appendix we present proofs for Lemma 24 and Lemma 25 for completeness. We remark that we did not make any effort to optimize the soundness/completeness constants s and c in our construction; any universal constant suffices in our applications.[16]

We start with Lemma 24 (restated below).

▶ **Reminder of Lemma 24** (3-query PCPP with perfect completeness)**.** *For any constant $\delta > 0$ there is a constant $0 < s < 1$, such that there is a PCP of proximity system for Circuit-Eval with proximity δ, soundness s, random bits $r = O(\log n)$, and query complexity $q = 3$. Moreover, the system satisfies two additional properties:*

(1) *Given the random coins, the verifier simply computes an OR on these 3 queried bits or their negations, and accepts if the OR is true.*

(2) *Given the pair $(C, w) \in$ Circuit-Eval, we can construct a proof π in $\mathrm{poly}(|C| + |w|)$ time that makes $V(C)$ accept with probability 1.*

Proof. By Lemma 22, there is a PCP of proximity system V for Circuit-Eval with proximity δ, soundness $s = 1/2$, number of random bits $r = O(\log n)$ and query complexity $q = O_\delta(1)$.

Let the circuit be C. Suppose we are given random bits $R \in \{0,1\}^r$, so that $V(C)$ queries positions $k_1 = k_1(C, R), k_2 = k_2(C, R), \ldots, k_q = k_q(C, R)$ of the oracle $z = w \circ \pi$, computes a predicate $P = P(C, R)$ on these bits, and outputs $P(z_{k_1}, z_{k_2}, \ldots, z_{k_q})$.

We construct a new PCP of proximity system V' as follows. Fix the circuit C. and let $P_R = P(C, R)$. Slightly abusing notation, we let x_j denote the bit z_{k_j}. P_R can be computed by a circuit D_R of $O_q(1)$ size; therefore P_R can computed in size S for a universal constant S only depending on q. We can construct a group of auxiliary variables $\{y_{R,\ell}\}_{\ell \in [S]}$, and a group of constraints $\{F_{w,\ell}\}_{\ell \in [S]}$, where each constraint is an OR of three bits (or their negations) from the x_j's and $y_{R,\ell}$'s, such that $P_w(x_1, x_2, \ldots, x_q) = 1$ if and only if there exists an assignment to the $y_{R,\ell}$'s such that all constraints $F_{R,\ell}$'s are satisfied.

The verifier $V'(C)$ treats its oracle as three parts. The first two parts are w and π (where π is supposed to be a proof for $V(C)$), while the third part π_y is supposed to contain assignments to all $y_{R,\ell}$'s for all $R \in \{0,1\}^r$ and $\ell \in [S]$. $V'(C)$ first tosses r random coins to get a random string $R \in \{0,1\}^r$, then tosses $\log(S)$ more coins to pick a random integer $\ell \in [S]$. Then $V'(C)$ queries the 3 bits appearing in the constraint $F_{R,\ell}$, and accepts if and only if the constraint is satisfied by those 3 bits. We denote its proof to be $\pi' = (\pi, \pi_y)$.

We claim that $V'(C)$ is a correct PCP of Proximity system. If $(C, w) \in$ Circuit-Eval, let π be a proof such that $V(C)$ accepts $w \circ \pi$ with probability 1. Then by our construction of $V'(C)$, there is a π_y such that $V'(C)$ accepts $w \circ (\pi \circ \pi_y)$ with probability 1.

[16] Here we are actually composing the PCPP from [11] with some trivial PCPP constructions for constant-size functions. There are much better constructions, see e.g. [18].

Otherwise, suppose w is δ-far from the set $\{z : C(z) = 1\}$. Then for any proof π, $V(C)$ accepts $(w \circ \pi)$ with probability at most $1/2$. This means for all additional proofs π_y, at least a $1/2$-fraction of $R \in \{0,1\}^r$ are such that at least one constraint from $\{F_{R,\ell}\}_{\ell \in [S]}$ is not satisfied by $w \circ (\pi \circ \pi_y)$. Therefore, $V'(C)$ rejects with probability at least $1/2 \cdot 1/S = \Omega_q(1) = \Omega_\delta(1)$, which completes the proof. ◀

In order to get a 2-query PCP of proximity system from the above, we use the following classical gadget by Garey, Johnson, and Stockmeyer [20], originally used to prove the NP-hardness of MAX-2-SAT.

▶ **Lemma 47.** *Let X_1, X_2, X_3 and Y be 4 Boolean variables. Consider the following 10 constraints:*

$$X_1, X_2, X_3, \neg X_1 \vee \neg X_2, \neg X_2 \vee \neg X_3, \neg X_3 \vee \neg X_1,$$
$$Y, X_1 \vee \neg Y, X_2 \vee \neg Y, X_3 \vee \neg Y.$$

If $X_1 \vee X_2 \vee X_3$, then there exists an assignment to Y such that 7 of the above constraints are satisfied. Otherwise, all assignments to Y satisfy at most 6 of the above constraints.

▶ **Reminder of Lemma 25** (2-query PCPP with constant completeness/soundness gap)**.** *For any constant $\delta > 0$ there two constants $0 < s < c < 1$, such that there is a PCP of proximity system for Circuit-Eval with proximity δ, soundness s, completeness c, number of random bits $r = O(\log n)$ and query complexity $q = 2$. Moreover, the system satisfies two additional properties:*

(1) *Given the random coins, the verifier computes an OR on the 2 queried bits or their negations, and accepts iff the OR is true.*

(2) *Given the pair $(C, w) \in$ Circuit-Eval, a proof π can be constructed in $\mathrm{poly}(|C| + |w|)$ time that makes $V(C)$ accept with probability at least c.*

Proof. By Lemma 24, there is a PCP of proximity system V for Circuit-Eval with proximity δ, soundness $s = s(\delta) < 1$, number of random bits $r = O(\log n)$ and query complexity $q = 3$. The verifier computes an OR on these 3 queried bits or their negations, and accepts if it is true.

Let the circuit be C. We begin as in the previous proof. Suppose we have randomness $R \in \{0,1\}^r$, and $V(C)$ queries positions $k_1 = k_1(C,R), k_2 = k_2(C,R), k_3 = k_3(C,R)$ of the oracle $z = w \circ \pi$, computes a predicate $P = P(C,R)$ on these bits, then outputs $P(z_{k_1}, z_{k_2}, z_{k_3})$. Slightly abusing notation, we use x_j to denote the bit z_{k_j}. By Lemma 24, we can assume

$$P(x_1, x_2, x_3) = \vee_{j \in [3]}(x_j \oplus b_j),$$

where $b_j = b_j(C,R)$ is whether it negates the bit x_j.

Our new PCP of proximity system V' works as follows. Fix the circuit C and let $P_R = P(C,R)$. By Lemma 47, we can construct an auxiliary variable y_R and a group of constraints $\{F_{R,\ell}\}_{\ell \in [10]}$, each is an OR of 2 bits (or their negations) from x_j's and y_R[17] such that if $P_w(x_1, x_2, x_3) = 1$ then there is an assignment to the y_R such that 7 constraints from $\{F_{R,\ell}\}_{\ell \in [10]}$ are satisfied; otherwise, for all assignments to y_R, at most 6 constraints from $\{F_{R,\ell}\}_{\ell \in [10]}$ are satisfied.

[17] In Lemma 47, constraint X_i can be written as $X_i \vee X_i$, which is an OR of 2 bits.

As in the 3-query PCPP, $V'(C)$ treats its oracle as three parts: the first two are w and π (π is intended to be a proof in $V(C)$), and the third part π_y is intended to contain assignments to all y_R's, for all $R \in \{0,1\}^r$. Our $V'(C)$ first tosses r random coins to get $R \in \{0,1\}^r$, then tosses $O(1)$ more coins to pick a random integer $\ell \in [10]$. Then it simply queries the 2 bits appearing in the constraint $F_{R,\ell}$, and accepts iff that constraint is satisfied. We denote its proof to be $\pi' = (\pi, \pi_y)$.

Let us argue $V'(C)$ satisfies our requirement. If $(C, w) \in$ Circuit-Eval, let π be a proof such that $V(C)$ accepts $w \circ \pi$ with probability 1. Then by our construction of $V'(C)$, there is a π_y such that $V'(C)$ accepts $w \circ (\pi \circ \pi_y)$ with probability at least $7/10$.

Now suppose w is δ-far from the set $\{z : C(z) = 1\}$. Then for all proofs π, $V(C)$ accepts $(w \circ \pi)$ with probability at most s. This means that for any additional proof π_y, there is at most an s-fraction of $R \in \{0,1\}^r$ such that 7 constraints from $\{F_{R,\ell}\}_{\ell \in [S]}$ are satisfied by $w \circ (\pi \circ \pi_y)$; for the remaining R's, at most 6 constraints from $\{F_{R,\ell}\}_{\ell \in [S]}$ are satisfied. Therefore, $V'(C)$ accepts with probability at most $s \cdot 7/10 + (1-s) \cdot 6/10 < 7/10$, which completes the proof. ◀

B Proofs for THR ⊆ DOR ∘ ETHR and $\oplus_k$ ∘ THR ∘ THR ⊆ THR ∘ THR

Here we present an alternative proof that THR ⊆ DOR ∘ ETHR, which has a better weight dependence than prior work [24] and is arguably simpler. We first give a construction for the special case when all the weights and the threshold value are non-negative. Then we show that the general case can be easily reduced to this case.

▶ **Lemma 48.** *Let G be a THR gate on n bits defined as $G(x) := [\sum_{i=1}^n w_i \cdot x_i > T]$, such that all w_i's and T are integers in $[0, 2^L - 1]$ for some $L \in \mathbb{N}$. Then G can be written as a DOR of $O(n \cdot L)$ many ETHR gates, each with weights and threshold from $[0, 2^{L+1} - 1]$.*

Proof. For each weight $w_i \in [0, 2^L - 1]$, write it in its binary representation

$$w_{i,L}, w_{i,L-2}, \ldots, w_{i,1} \in \{0,1\}^L,$$

such that

$$w_i = \sum_{j=1}^{L} 2^{j-1} \cdot w_{i,j}.$$

In this way, we can view w as a Boolean matrix from $\{0,1\}^{n \times L}$. For each position $(a,b) \in [n] \times [L]$, we build a partial matrix $w^{(a,b)}$ as follows: for $(i,j) \in [n] \times [L]$,

$$w^{(a,b)}_{i,j} = \begin{cases} w_{i,j} & (j > b) \text{ or } ((j = b) \text{ and } (i \geq a)) \\ 0 & \text{otherwise} \end{cases}$$

That is, $w^{(a,b)}$ is the sub-matrix of w, consisting of entries which are either to the right of (a,b), or directly above (a,b). (We number the rows of the matrix from bottom to top, and the columns of the matrix from left to right.)

Given $x \in \{0,1\}^n$, we define

$$w^{(a,b)} \cdot x := \sum_{i=1}^{n} \left(\sum_{j=1}^{L} 2^{j-1} \cdot w^{(a,b)}_{i,j} \right) \cdot x_i.$$

That is, we treat each row of $w^{(a,b)}$ as the L-bit binary representation of the corresponding weight on x_i. By definition, we have $w^{(1,1)} \cdot x = w \cdot x = \sum_{i=1}^{n} w_i \cdot x_i$.

Now, fix $x \in \{0,1\}^n$, and consider the sequence S, defined as:

$$w^{(n,L)} \cdot x, w^{(n-1,L)} \cdot x, \ldots, w^{(1,L)} \cdot x, w^{(n,L-1)} \cdot x, \ldots, w^{(1,L-1)} \cdot x, \ldots, w^{(n,1)} \cdot x, \ldots, w^{(1,1)} \cdot x.$$

By definition of $w^{(a,b)}$, we are including 1-entries of the matrix w one-by-one, hence the sequence S is non-decreasing (and begins with 0).

Suppose $w \cdot x = w^{(1,1)} \cdot x > T$. Then there must be a *unique* position $(a,b) \in [n] \times [L]$ such that $w^{(a,b)} \cdot x$ is the first value in the sequence S which is greater than T. For each $(a,b) \in [n] \times [L]$, we will use an ETHR gate $E^{(a,b)}$ to specify the condition that $w^{(a,b)} \cdot x$ is the first value greater than T from S. Then when $G(x)$ is true, exactly one of the $E^{(a,b)}(x)$'s is true, and when $G(x)$ is false, all of the $E^{(a,b)}(x)$'s are false.

To see that an ETHR gate suffices, we observe that $w^{(a,b)} \cdot x$ is the first value greater than T from the sequence S, if and only if the following conditions hold:

1. $w^{(a,b)} \cdot x > T$ *(it is greater than T)*,
2. $(w_{a,b} = 1) \wedge (x_a = 1)$ *(it is bigger than the previous value)*, and
3. $w^{(a,b)} \cdot x - 2^{b-1} \leq T$ *(the previous value is no greater than T)*.

We crucially observe that $w^{(a,b)} \cdot x$ is a multiple of 2^{b-1}. In the matrix $w^{(a,b)}$, we only include nonzero $w_{i,j}$'s where $j \geq b$. Thus in $w^{(a,b)} \cdot x$, every 1 in x is getting multiplied by a power of two which is at least 2^{b-1}.

By division, $T = 2^{b-1} \cdot T_b + T_r$, for some $0 \leq T_r < 2^{b-1}$ and $T_b > 0$. Then $w^{(a,b)} \cdot x > T$ if and only if $w^{(a,b)} \cdot x \geq 2^{b-1} \cdot (T_b + 1)$. Furthermore, $w^{(a,b)} \cdot x - 2^{b-1} \leq T$ if and only if $w^{(a,b)} \cdot x \leq 2^{b-1} \cdot (T_b + 1)$. Therefore, the above conditions are equivalent to

1. $(w_{a,b} = 1) \wedge (x_a = 1)$, and
2. $w^{(a,b)} \cdot x = 2^{b-1} \cdot (T_b + 1)$.

Now all these conditions are linear equations, so we can define an ETHR function $E^{(a,b)}$ that checks all of them. In particular, set $E^{(a,b)}$ to be the constant function $\mathbf{0}$ if $w_{a,b} = 0$; otherwise set

$$E^{(a,b)}(x) := \left[(2 \cdot w^{(a,b)} \cdot x) + x_a = 2^b \cdot (T_b + 1) + 1 \right].$$

This completes the proof. ◀

Now we reduce the general case to the non-negative weights and thresholds case, and complete the reduction from THR to $\mathsf{DOR} \circ \mathsf{ETHR}$.

▶ **Lemma 49.** *Let G be a THR gate on n bits, $G(x) := [\sum_{i=1}^{n} w_i \cdot x_i > T]$, such that all w_i's and T are integers from $[-W, W]$ for some $W \in \mathbb{N}$. Then G can be written as a DOR of $O(n \cdot \log W)$ many ETHR gates, each with weights and threshold from $[-\Theta(W), \Theta(W)]$.*

Proof. We start by defining n new variables $z_1, z_2, \ldots, z_n \in \{0,1\}^n$. Set $z_i := x_i$ if $w_i \geq 0$, and $z_i := 1 - x_i$ otherwise. Letting $S = \{i : w_i \geq 0\}$, we have

$$\begin{aligned} \sum_{i=1}^{n} w_i \cdot x_i &= \sum_{i \in S} w_i \cdot z_i + \sum_{i \notin S} w_i \cdot (1 - z_i) \\ &= \sum_{i \in S} w_i \cdot z_i + \sum_{i \notin S} -w_i \cdot z_i + \sum_{i \notin S} w_i. \end{aligned}$$

Let $\widehat{w}_i = |w_i|$, and $\widehat{T} = T - \sum_{i \notin S} w_i$. Observe that

$$\left[\sum_{i=1}^{n} w_i \cdot x_i > T \right] \Leftrightarrow \left[\sum_{i=1}^{n} \widehat{w}_i \cdot z_i > \widehat{T} \right].$$

If $\widehat{T}$ is negative, then $G(x) = 1$ on all Boolean inputs x (since all $\widehat{w}_i$'s are non-negative, and the z_i are Boolean) and we are done. Otherwise, we can apply Lemma 48 with $\widehat{G}(z) := \left[\sum_{i=1}^{n} \widehat{w}_i \cdot z_i > \widehat{T}\right]$. Substituting each z_i by $1 - x_i$, we obtain the desired DOR decomposition for $G(x)$. ◀

▶ **Lemma 50.** *Let k be a constant, a $\oplus_k \circ$ THR $\circ$ THR circuit of $s = s(n)$ size on n bits is equivalent to a THR $\circ$ THR circuit of $s^{O(k)}$ size. Moreover, the corresponding THR $\circ$ THR circuit can be constructed deterministically in $s^{O(k)}$ time.*

Proof. First, by Lemma 49, the given $\oplus_k \circ \mathsf{THR} \circ \mathsf{THR}$ circuit can be transformed into a $\oplus_k \circ \mathsf{THR} \circ \mathsf{ETHR}$ circuit C of $t = \mathrm{poly}(s)$ size.

Let $C_1, C_2, \ldots, C_k$ be the $\mathsf{THR} \circ \mathsf{ETHR}$ subcircuits of C. For each C_i, let $E_{i,1}, \ldots, E_{i,t}$ be its ETHR gates, $w_{i,1}, \ldots, w_{i,t}$ be the corresponding weights in the output threshold function, and T_i be the threshold value of the output threshold function. We have

$$C_i(x) := \left[\sum_{j=1}^{t} w_{i,j} \cdot E_{i,j}(x) > T_i\right].$$

By slightly perturbing the $w_{i,j}$'s and T_i, we can ensure that $\sum_{j=1}^{t} w_{i,j} \cdot E_{i,j}(x) - T_i$ is never equal to 0, over all $x \in \{0,1\}^n$. Next, we define

$$F(x) = \left[\prod_{i=1}^{k} \left(T_i - \sum_{j=1}^{t} w_{i,j} \cdot E_{i,j}(x)\right) < 0\right]. \tag{8}$$

Noting that $C_i(x) = 1$ if and only if $T_i - \sum_{j=1}^{t} w_{i,j} \cdot E_{i,j}(x) < 0$, we observe that $F(x) = 1$ when an odd number of $C_i(x)$'s are 1, and $F(x) = 0$ otherwise. Therefore, F computes the same function as the original circuit C.

Finally, expanding the product of k sums in (8) into a sum of $s^{O(k)}$ products, and recalling that $\mathsf{AND} \circ \mathsf{ETHR} \subseteq \mathsf{ETHR}$ (Proposition 18), F can be written as a $\mathsf{THR} \circ \mathsf{ETHR}$ circuit. Converting this back to a $\mathsf{THR} \circ \mathsf{THR}$ circuit (Proposition 18), the proof is complete. ◀

C More Applications of Structure Lemmas for THR ∘ THR Circuits

Here we discuss more applications of Lemma 16 and Lemma 17.

C.1 Generalization to Threshold Circuits of Constant Depth

Lemma 17 generalizes readily to threshold circuits of any constant depth. In the following LT_d denotes threshold circuits of depth-d, while $\widehat{\mathsf{LT}}_d$ denotes depth-d majority circuits (see Section 2.1 for formal definitions).

▶ **Corollary 51.** *Let n be number of inputs and $s = s(n)$ be a size parameter. Let $\varepsilon \in \left(\frac{\log s}{n}, 1\right)$ and d be a constant. For $s = 2^{o(n)}$, every s-size LT_d circuit is equivalent to a DOR $\circ$ $\widehat{\mathsf{LT}}_d$ circuit such that:*

- *The top DOR gate has $2^{O(\varepsilon \cdot n)}$ fan-in.*
- *Each sub $\widehat{\mathsf{LT}}_d$ circuit has size $O\left(s^{O(1/\varepsilon)}\right)$.*

Proof. We apply Lemma 17 to the top 2 layers, and then apply item (5) of Proposition 18 recursively to obtain an equivalent $\mathsf{DOR} \circ \widehat{\mathsf{LT}}_d$ circuit. ◀

C.2 A Structure Lemma for Polynomial Threshold Functions

Our ideas can also be used to derive a structure lemma for polynomial threshold functions of degree k, i.e., $\mathsf{THR} \circ \mathsf{AND}_k$ circuits:

▶ **Corollary 52.** *Let n be number of inputs and $s = s(n)$ be a size parameter. Let $\varepsilon \in \left(\frac{\log s}{n}, 1\right)$ and k be a constant. Assuming $s = 2^{o(n)}$, an s-size $\mathit{THR} \circ \mathit{AND}_k$ circuit is equivalent to a $\mathit{DOR} \circ \mathit{MAJ} \circ \mathit{AND}_{2k}$ circuit such that:*

- *The top DOR gate has $2^{O(\varepsilon \cdot n)}$ fan-in.*
- *Each sub $\mathit{MAJ} \circ \mathit{AND}_{2k}$ circuit has size $O\left(s^{O(1/\varepsilon)}\right)$.*

The above still holds if we replaced both AND_k and AND_{2k} by unbounded fan-in AND gates.

Proof. We simply apply Lemma 43 and Lemma 45, and merge each $\mathsf{AND}_2 \circ \mathsf{AND}_k$ subcircuits into a single AND_{2k} gate. ◀

That is, every polynomial threshold function of degree k with arbitrary weights can be simulated by a *subexponential-size* disjoint OR of polynomial threshold functions of degree $2k$ with small weights.

The following corollary follows from that the SAT problem for $\mathsf{THR} \circ \mathsf{AND}_k$ circuits is equivalent to the *weighted* MAX-k-SAT problem (given a CNF formula φ with weights on each clause, find an assignment satisfying clauses of maximum total weight), and that SAT for $\mathsf{MAJ} \circ \mathsf{AND}_{2k}$ is equivalent to the (unweighted) MAX-$2k$-SAT problem.

▶ **Corollary 53.** *For any integer k, if there is a $2^{(1-\Omega(1))n}$ time algorithm for polynomial size unweighted MAX-$2k$-SAT, then so does polynomial size weighted MAX-k-SAT.*[18]

To prove the above corollary, we need the following folklore lemma, which helps us to transform between $\mathsf{MAJ} \circ \mathsf{AND}$ circuits and $\mathsf{MAJ} \circ \mathsf{OR}$ circuits.

▶ **Lemma 54.** *Let $x = x_1, x_2, \ldots, x_k$ be the inputs, there are k OR functions $O_1, O_2, \ldots, O_k$ on the inputs (or their negations) such that:*

$$\mathit{AND}(x) = \left(\sum_{i=1}^{k} O_i(x)\right) - (k-1).$$

Proof. We define

$$O_i(x) := \left(\bigvee_{j=1}^{i-1} \neg x_j\right) \vee x_i.$$

That is, $O_i(x) = 0$ if and only if the first $i-1$ bits are 1, and the i-th bit is 0. Now, note that if $\mathsf{AND}(x) = 1$, then all bits are 1, which means all $O_i(x)$'s are 1. When $\mathsf{AND}(x) = 0$, let i be the index of the first 0-bit, it is easy to see that $O_i(x) = 0$ and all other $O_j(x)$'s are 1, and hence $\sum_{i=1}^{k} O_i(x) = k - 1$. ◀

Proof of Corollary 53. We can use Lemma 54 to transform the bottom AND gates to OR gates for $\mathsf{THR} \circ \mathsf{AND}$ and $\mathsf{MAJ} \circ \mathsf{AND}$ circuits. From there, the proof is the same as of Theorem 3. ◀

[18] We assume the weights are at most $2^{\mathrm{poly}(n)}$ for making the input polynomial size.

Universality of EPR Pairs in Entanglement-Assisted Communication Complexity, and the Communication Cost of State Conversion

Matthew Coudron
Institute for Quantum Computing, University of Waterloo, Canada
mcoudron@uwaterloo.ca

Aram W. Harrow
Center for Theoretical Physics, MIT, Cambridge, MA, USA
https://web.mit.edu/aram/www/
aram@mit.edu

Abstract

In this work we consider the role of entanglement assistance in quantum communication protocols, focusing, in particular, on whether the type of shared entangled state can affect the quantum communication complexity of a function. This question is interesting because in some other settings in quantum information, such as non-local games, or tasks that involve quantum communication between players and referee, or simulating bipartite unitaries or communication channels, maximally entangled states are known to be less useful as a resource than some partially entangled states. By contrast, we prove that the bounded-error entanglement-assisted quantum communication complexity of a partial or total function cannot be improved by more than a constant factor by replacing maximally entangled states with arbitrary entangled states. In particular, we show that every quantum communication protocol using Q qubits of communication and arbitrary shared entanglement can be ϵ-approximated by a protocol using $O(Q/\epsilon+\log(1/\epsilon)/\epsilon)$ qubits of communication and *only* EPR pairs as shared entanglement. This conclusion is opposite of the common wisdom in the study of non-local games, where it has been shown, for example, that the I3322 inequality has a non-local strategy using a non-maximally entangled state, which surpasses the winning probability achievable by any strategy using a maximally entangled state of any dimension [17]. We leave open the question of how much the use of a shared maximally entangled state can reduce the quantum communication complexity of a function.

Our second result concerns an old question in quantum information theory: How much quantum communication is required to approximately convert one pure bipartite entangled state into another? We give simple and efficiently computable upper and lower bounds. Given two bipartite states $|\chi\rangle$ and $|\upsilon\rangle$, we define a natural quantity, $d_\infty(|\chi\rangle, |\upsilon\rangle)$, which we call the ℓ_∞ Earth Mover's distance, and we show that the communication cost of converting between $|\chi\rangle$ and $|\upsilon\rangle$ is upper bounded by a constant multiple of $d_\infty(|\chi\rangle, |\upsilon\rangle)$. Here $d_\infty(|\chi\rangle, |\upsilon\rangle)$ may be informally described as the minimum over all transports between the log of the Schmidt coefficients of $|\chi\rangle$ and those of $|\upsilon\rangle$, of the maximum distance that any amount of mass must be moved in that transport. A precise definition is given in the introduction. Furthermore, we prove a complementary lower bound on the cost of state conversion by the ϵ-Smoothed ℓ_∞-Earth Mover's Distance, which is a natural smoothing of the ℓ_∞-Earth Mover's Distance that we will define via a connection with optimal transport theory.

2012 ACM Subject Classification Theory of computation → Quantum communication complexity; Theory of computation → Quantum information theory

Keywords and phrases Entanglement, quantum communication complexity

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.20

Related Version A full version is available at arXiv:1902.07699.

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka; Article No. 20; pp. 20:1–20:25

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

COMPUTATIONAL COMPLEXITY CONFERENCE

Funding *Matthew Coudron*: Supported at the IQC by Canada's NSERC and the Canadian Institute for Advanced Research (CIFAR), and through funding provided to IQC by the Government of Canada and the Province of Ontario.
Aram W. Harrow: NSF grants CCF-1452616, CCF-1729369, PHY-1818914 and ARO contract W911NF-17-1-0433.

1 Introduction

1.1 Entanglement-assisted communication complexity

Imagine that two cooperating players, Alice and Bob, are given the task of evaluating a function $f(x, y)$ $(x, y \in \{0, 1\}^n)$, where x is known only to Alice and y is known only to Bob. The communication complexity of f is the number of bits that Alice and Bob need to exchange in order to compute f. Popular variations of this framework include allowing a small probability of error, allowing qubits to be communicated instead of classical bits, and allowing extra resources such as shared randomness or entanglement.

In classical communication complexity, Newman's theorem [14] states that arbitrarily large amounts of shared randomness in a protocol can be replaced by a distribution with $O(\log(n/\epsilon))$ bits of entropy while only reducing the success probability of that protocol by ϵ. (Here n is the input size of each party.) Is there a quantum analogue to this result?

In one sense the answer is "no". Given a two-party entanglement-assisted protocol for, say, computing the value of some function, we cannot replace the shared entanglement with some different, less entangled, state, without causing large errors [10, 1]. It is an open question whether it is possible to replace a large entangled state with a less entangled one while also changing the communication protocol.

However, while it remains a challenge to characterize the *dimension* of shared entanglement required for optimal entanglement-assisted quantum communication protocols, in this work we show that the *type* of shared entanglement required by such protocols can be neatly characterized. In Theorem 1 below, we establish that the bounded-error entanglement-assisted quantum communication complexity of a partial or total function cannot be improved by more than a constant factor by replacing maximally entangled states with arbitrary entangled states. This is accomplished by constructing an explicit protocol which allows two parties, who only share maximally entangled states, to simulate any entanglement-assisted quantum communication task regardless of the shared state that that task originally required.

▶ **Theorem 1.** *Consider a quantum communication protocol $\mathcal{R}$ whose goal it is to compute a joint function $f(x, y) \in \{0, 1\}$. Suppose that $\mathcal{R}$ uses an arbitrary bipartite entangled state $|\psi\rangle^{AB}$ (of unbounded dimension), as well as Q qubits of communication total, in either direction (for sufficiently large $Q \geq 15$). Then, for every $\epsilon > 0$, there exists a quantum communication protocol $\mathcal{R}'$ which simulates $\mathcal{R}$ with error ϵ, while using only a maximally entangled state as an entangled resource (rather than $|\psi\rangle^{AB}$ or any other state), and using $O(Q/\epsilon + \log(1/\epsilon)/\epsilon)$ qubits of communication. Thus, if $\mathcal{R}$ computes f with error ϵ' it follows that $\mathcal{R}'$ computes f with error $\epsilon + \epsilon'$.*

Theorem 1 shows that, although the role of shared entanglement in quantum communication complexity is still not well understood, the *type* of shared entanglement does not drastically change communication complexity. This is true regardless of input size or promise, as long as we are in the constant-error regime and some communication is allowed between players (unlike, say, the simultaneous-message-passing model). This result sets quantum communication complexity apart from settings such as channel simulation [3], nonlocal

games [11, 16], unitary gate simulation [6], and communication tasks involving quantum communication between referees and players [12]. In each of those cases the ratio between the EPR-assisted costs and the (unrestricted) entanglement-assisted costs can be made arbitrarily large. This suggests that the role of shared entanglement in quantum communication complexity may be fundamentally different than in these other settings. Furthermore, the result achieved in Theorem 1 may be useful in future work attempting to further bound the role of entanglement in quantum communication complexity, as it restricts the problem to the case of shared EPR pairs, without loss of generality.

Previously it was known that a universal form of entanglement existed: the embezzling state [8]. Our Theorem 1 can be viewed as showing that these states can be replaced by the much simpler family of maximally entangled states.

It may be worth noting that the proof of Theorem 1 is nearly oblivious to the entanglement-assisted protocol being considered in the following sense: Given a protocol $\mathcal{P}$ using Q qubits of entanglement and a shared entangled state $|\psi\rangle$, we can replace $|\psi\rangle$ with a "consolidated" state ρ at the cost of error ϵ. Moreover, ρ can be prepared from a maximally entangled state using $O(Q/\epsilon + \log(1/\epsilon)/\epsilon)$ communication. Taking ϵ constant implies that the EPR-assisted communication complexity of a function is at most $O(1)$ times the (unrestricted) entanglement-assisted communication complexity of that function. It was not necessary to modify the protocol $\mathcal{P}$ to achieve this result, except to pre-compose it with a pre-processing protocol which starts with only EPR pairs, and prepares the state ρ using only $O(Q/\epsilon + \log(1/\epsilon)/\epsilon)$ communication. $\mathcal{P}$ can then be run on ρ directly. Such a protocol-agnostic preprocessing should not be taken for granted, since it is known that reducing the number of EPR pairs may in some cases require more than just pre-processing [10, 1].

1.2 Communication cost of state transformations

Our second contribution, which is related at the level of techniques to Theorem 1, is to provide upper and lower bounds for an old quantity studied in quantum information theory, the communication cost of state transformation.

Suppose that $|\chi\rangle^{AB}$ and $|\nu\rangle^{AB}$ are bipartite pure quantum states, with vectors of Schmidt coefficients denoted respectively by χ and ν. In this setting it is known that $|\chi\rangle$ can be exactly converted into $|\nu\rangle$ using LOCC if and only if χ is majorized by ν [15]. But the communication cost of this transformation is known only in a few special cases. If $|\chi\rangle = |\chi_0\rangle^{\otimes n}$ and $|\nu\rangle = |\nu_0\rangle^{\otimes n}$ for some states $|\chi_0\rangle, |\nu_0\rangle$, then this cost is $O(\sqrt{n})$ or less in some special cases (e.g. $|\nu_0\rangle$ is maximally entangled). More generally there is, in principle, an exact characterization of the communication cost (either LOCC, or quantum communication) of state transformation using the Schubert calculus due to Daftuar and Hayden [4], but in practice it is difficult to extract concrete bounds from their main theorem.

In this work we identify a simple and efficiently computable quantity, which we call the ℓ_∞ Earth Mover's (or Wasserstein) Distance, which tells us approximately how much quantum communication is required to transform $|\chi\rangle$ to $|\nu\rangle$. Given its simple form, we believe that this quantity may be a useful tool in quantum information theory.

▶ **Definition 2** (ℓ_∞ Earth Mover's Distance). *Let* $|\chi\rangle^{AB} = \sum_{i\in X} \sqrt{\chi_i}\,|i\rangle^A \otimes |i\rangle^B$ *and* $|\upsilon\rangle^{AB} = \sum_{j\in Y} \sqrt{\upsilon_j}\,|j\rangle^A \otimes |j\rangle^B$ *be two states. We define* $d_\infty(|\chi\rangle, |\upsilon\rangle)$ *to be the* ℓ_∞ *Earth Mover's distance between* $|\chi\rangle$ *and* $|\upsilon\rangle$*, which is equal to the minimum* $\mu \geq 0$ *for which there exists a joint distribution* $\omega(x, y) : X \times Y \to \mathbb{R}_{\geq 0}$ *such that:*

- $\sum_{j\in Y} \omega(i, j) = \chi_i \ \forall i \in X$
- $\sum_{i\in X} \omega(i, j) = \upsilon_j \ \forall j \in Y$
- $\omega(i, j) = 0$ *whenever* $|\log(\chi_i) - \log(\upsilon_j)| > \mu$

We can think of χ as corresponding to placing χ_i mass at position $\log(\chi_i)$ for each i, and similarly for υ. Then $d_\infty(|\chi\rangle, |\upsilon\rangle)$ is the ℓ_∞ EMD (Earth Mover's distance) between these distributions.

In Section 4 we will show that this quantity gives an intuitive upper bound on the amount of quantum communication required to transform one bipartite shared state into another. In particular we prove the following theorem.

▶ **Theorem 3.** *Let $|\chi\rangle^{AB}$ and $|\upsilon\rangle^{AB}$ be two bipartite shared states. There is a protocol $\mathcal{M}_{\chi\to\upsilon}$ which can prepare $|\upsilon\rangle$ from $|\chi\rangle$, using only $4\lceil d_\infty(|\chi\rangle, |\upsilon\rangle)\rceil + 8$ qubits of communication.*

In Section 3 we establish a complementary lower bound, showing that a "ϵ-smoothed" version of the ℓ_∞ Earth Mover's Distance, denoted by $d^\epsilon_\infty(|\chi\rangle, |\upsilon\rangle)$, gives a lower bound on the cost of state transformation. That is:

▶ **Theorem 4.** *Given any two bipartite shared states $|\psi\rangle^{AB} = \sum_i \sqrt{\psi_i}\,|i\rangle^A \otimes |i\rangle^B$ and $|\phi\rangle^{AB} = \sum_i \sqrt{\phi_i}\,|i\rangle^A \otimes |i\rangle^B$, shared between two parties A and B, together with a unitary $U_\mathcal{P}$ which can be performed on the state $|\psi\rangle^{AB}$ via a quantum communication protocol $\mathcal{P}$, that uses Q qubits of communication between A and B, we have that, for every ϵ:*

$$\left|\langle\phi|^{AB} U_\mathcal{P} |\psi\rangle^{AB}\right| \leq 1 - \frac{1}{4}\epsilon^2 + 24 \cdot 2^{-\frac{1}{2}(d^\epsilon_\infty(|\psi\rangle,|\phi\rangle)-3Q)}$$

In words: If two shared states cannot be brought within small ℓ_∞ Earth Mover's Distance of each other by moving an ϵ quantity of mass of their Schmidt coefficients, then they also cannot be brought closer than $1-O(\epsilon^2)$ fidelity with each other without using $\Omega(d^\epsilon_\infty(|\psi\rangle, |\phi\rangle))$ qubits of communication (for sufficiently large values of $d^\epsilon_\infty(|\psi\rangle, |\phi\rangle)$). Thus, the ϵ-smoothed ℓ_∞ Earth Mover's Distance provides a lower bound on the communication cost of state conversion. On the other hand, from the definition of $d^\epsilon_\infty(|\psi\rangle, |\phi\rangle)$, stated in Definition 13, we note here that one can use Theorem 3 to move $|\psi\rangle$ to within $1-\epsilon$ fidelity of $|\phi\rangle$ using only $O(d^\epsilon_\infty(|\psi\rangle, |\phi\rangle))$ qubits of communication. To do this, omit the ϵ mass of Schmidt coefficients on which the two states have large ϵ-smoothed ℓ_∞ distance, and apply Theorem 3 as one would do with the regular ℓ_∞ Earth Mover's Distance. In this sense $d^\epsilon_\infty(|\psi\rangle, |\phi\rangle)$ gives both an upper and lower bound on the communication cost of state conversion.

To put these bounds in context: One could consider entanglement concentration and dilution to be the starting point for the study of state conversion. The original paper on entanglement concentration and dilution [2] concerned the many-copy limit and did not attempt to bound the amount of classical communication used. The first time the classical communication cost of state conversion was considered explicitly seems to have been in [13], which could be said to establish a version of our upper bound in the case where the starting state is maximally entangled. (Their result is not quite that general but contains many of the key ideas.) A version of our lower bound was established, again for the case of starting with maximally entangled states, in [7, 9]. These lower bounds could be applied to general state conversion but relied on Rènyi entropy inequalities that are clearly not tight in many cases. Finally, as noted earlier, a full characterization of the communication cost of general state conversion was given in [4] but the resulting formula is complicated and there is not an efficient algorithm known to evaluate it.

We conclude the section with two remarks about notation.

▶ Remark 5. In theorem statements above, and where appropriate, we have made use of superscripts A and B, as in $|\psi\rangle^{AB} = \sum_i \sqrt{\psi_i}\,|i\rangle^A \otimes |i\rangle^B$ to explicitly denote the two halves of the bipartite division of a state. However, since all of the shared entangled states considered in this paper are bipartite, and since the two components of the bipartite division are generally clear from context, we will usually omit this notation.

▶ Remark 6. When considering a bipartite state $|\psi\rangle$, we will assume that the state has a Schmidt decomposition of the form $|\psi\rangle = \sum_i \sqrt{\psi_i}\,|i\rangle \otimes |i\rangle$ across the implicit bipartite division. This is done in the theorem statements above and everywhere in the paper. We can assume this WLOG because any state that has the same Schmidt coefficients as $|\psi\rangle$ can be moved to this canonical form (and vice versa) using only local unitary transformations, which can be implemented with no quantum communication between the two components of the bipartite division. Thus our analysis of communication costs is unaffected by assuming WLOG that, in any quantum communication protocol, shared entangled states start and end in this form.

▶ Remark 7. Given a quantum state $|\nu\rangle$ in $\mathcal{H}_A \otimes \mathcal{H}_B$, we will use $rk_{Schmidt}(|\nu\rangle)$ to denote the Schmidt rank of $|\nu\rangle$ across the bipartite division between $\mathcal{H}_A$ and $\mathcal{H}_B$.

2 Entanglement-Assisted Communication Complexity

In this section we will discuss the proof of our main result, Theorem 1, which shows that arbitrary entanglement-assisted quantum communication protocols can be simulated by quantum communication protocols that use only the maximally entangled state as an entangled resource. A basic fact we will need is that two bipartite pure states which are sufficiently different in the distribution of mass across their Schmidt coefficients must be nearly orthogonal. This fact is stated for our specific purposes in Lemma 9 below. Crucially, such states *remain* nearly orthogonal even after one of them is acted on by any unitary which can be implemented with a small amount of quantum communication, as we detail in Lemma 8.

▶ **Lemma 8.** *Given two quantum states $|\psi\rangle$ and $|\nu\rangle$ on $\mathcal{H}_A \otimes \mathcal{H}_B$, such that the Schmidt coefficients of ψ are upper bounded by $\lambda_{\max}$, and those of ν are upper bounded by $\nu_{\max}$, and further given a unitary transformation $\mathcal{U}$ on $\mathcal{H}_A \otimes \mathcal{H}_B$ which can be implemented using at most Q qubits of communication between the $\mathcal{H}_A$ and $\mathcal{H}_B$ components of the Hilbert space, it follows that:*

$$|\langle\psi|\mathcal{U}|\nu\rangle| \leq 2^{\frac{3}{2}Q} \cdot rk_{Schmidt}(|\psi\rangle)\sqrt{\lambda_{\max}\nu_{\max}}$$

Proof. If $\mathcal{U}$ is a unitary transform using Q qubits of communication, then $rk_{Schmidt}(\mathcal{U}|\nu\rangle) \leq 2^Q rk_{Schmidt}(|\nu\rangle)$ [9]. We also know that the Schmidt coefficients of $\mathcal{U}|\nu\rangle$ are bounded above by $2^Q\nu_{max}$ [9]. The desired result now follows by Lemma 9. ◀

▶ **Lemma 9.** *Given two quantum states $|\psi\rangle$ and $|\nu\rangle$ on $\mathcal{H}_A \otimes \mathcal{H}_B$, such that the Schmidt coefficients of ψ are upper bounded by $\lambda_{\max}$, and those of ν are upper bounded by $\nu_{\max}$, we have:*

$$|\langle\psi|\nu\rangle| \leq rk_{Schmidt}(|\psi\rangle)\sqrt{\lambda_{\max}\nu_{\max}}$$

Proof. For brevity let $r = rk_{Schmidt}(|\psi\rangle)$. Schmidt decompose $|\psi\rangle$ and $|\nu\rangle$ as $|\psi\rangle = \sum_{i=0}^{r-1}\sqrt{\lambda_i}\,|i\rangle_A \otimes |i\rangle_B$, as $|\nu\rangle = \sum_j \sqrt{\nu_j}\,|j\rangle_A \otimes |j\rangle_B$. Define the matrix $M_\nu = \sum_j \sqrt{\nu_j}\,|j\rangle_A\,\langle j|_B^*$, and note that

$$\begin{aligned}\langle\psi|\nu\rangle &= \sum_{i=0}^{r-1}\sum_{j}\sqrt{\lambda_i\nu_j}\,\langle i_A|j_A\rangle\cdot\langle i_B|j_B\rangle \\ &= \sum_{i=0}^{r-1}\sqrt{\lambda_i}\,\langle i_A|\left(\sum_j\sqrt{\nu_j}\,|j\rangle_A\,\langle j|_B^*\right)|i_B^*\rangle \\ &= \sum_{i=0}^{r-1}\sqrt{\lambda_i}\,\langle i_A|\,M_\nu\,|i_B^*\rangle\end{aligned}$$

Now, by definition of a Schmidt Decomposition, we know that the maximum singular value of M_ν is $\sqrt{\nu_{max}}$. Thus, for all i we have that $|\,\langle i_A|\,M_\nu\,|i_B^*\rangle\,| \leq \sqrt{\nu_{max}}$ (since $|i_A\rangle$ and $|i_B\rangle$ are normalized vectors by definition). It then follows that:

$$|\,\langle\psi|\nu\rangle\,| \leq r\sqrt{\lambda_{\max}\nu_{\max}} = rk_{Schmidt}(|\psi\rangle)\sqrt{\lambda_{\max}\nu_{\max}}$$ ◀

Although Theorem 1 is the main result of this work, the proof is too long to fit in a 10 page abstract. Therefore will now give a brief, intuitive outline of the proof of Theorem 1, restated below for the reader's convenience, and include the complete proof in Section A of the Appendix.

▶ **Theorem** (Restatement of Theorem 1). *Consider a quantum communication protocol $\mathcal{R}$ whose goal it is to compute a joint function $g(x,y) \in \{0,1\}$. Suppose that $\mathcal{R}$ uses an arbitrary bipartite entangled state $|\psi\rangle^{AB}$ (of unbounded dimension), as well as Q qubits of communication total, in either direction (for sufficiently large $Q \geq 15$). Then, for every $\epsilon > 0$, there exists a quantum communication protocol $\mathcal{R}'$ which simulates $\mathcal{R}$ with error ϵ, while using only a maximally entangled state as an entangled resource (rather than $|\psi\rangle^{AB}$ or any other state), and using $O(Q/\epsilon + \log(1/\epsilon)/\epsilon)$ qubits of communication. Thus, if $\mathcal{R}$ computes f with error ϵ' it follows that $\mathcal{R}'$ computes f with error $\epsilon + \epsilon'$.*

Proof Sketch of Theorem 1. Suppose that we have a communication protocol using Q qubits of communication and a pure entangled state $|\varphi\rangle$. If we can prepare $|\varphi\rangle$ from EPR pairs using $O(Q)$ qubits of communication then we are done. Thus we can assume that $|\varphi\rangle$ has entanglement spread that is $\gg Q$. This will be defined more precisely below (see also [9, 7, 5]) but roughly speaking it means that we can write $|\varphi\rangle$ as a superposition of varying numbers of EPR pairs, say from $E_{\min}$ to $E_{\max}$, with $E_{\max} - E_{\min} \gg Q$. (Technically we need to use Theorem 3 to show that with a small amount of communication $|\varphi\rangle$ can be mapped to a superposition of maximally entangled states of different sizes.)

For simplicity, suppose that $|\varphi\rangle = |\alpha\rangle + |\beta\rangle$ where $|\alpha\rangle, |\beta\rangle$ each have norm $1/\sqrt{2}$ and, up to normalization, $|\alpha\rangle$ is locally equivalent to $E_{\min}$ EPR pairs and $|\beta\rangle$ is locally equivalent to $E_{\max}$ EPR pairs. This difference in entanglement means that $|\alpha\rangle$ and $|\beta\rangle$ must be nearly orthogonal: specifically their overlap can be at most $1/\sqrt{2}^{E_{\max}-E_{\min}}$. Moreover, if we apply a unitary communication protocol $\mathcal{P}$ using Q qubits of communication to one of them, say β, then that will not be enough to bridge the gap. If we perform $\mathcal{P}$ and then measure the first qubit, this is equivalent to measuring the observable $\mathcal{P}^\dagger\sigma_z^{(1)}\mathcal{P}$, which conveniently is also a unitary using $2Q$ qubits of communication. The bias of the protocol (i.e. $\Pr[1] - \Pr[0]$) is then

$$\langle\varphi|\,\mathcal{P}^\dagger\sigma_z^{(1)}\mathcal{P}\,|\varphi\rangle = \langle\alpha|\,\mathcal{P}^\dagger\sigma_z^{(1)}\mathcal{P}\,|\alpha\rangle + \langle\beta|\,\mathcal{P}^\dagger\sigma_z^{(1)}\mathcal{P}\,|\beta\rangle + \tag{1}$$

$$\langle\alpha|\,\mathcal{P}^\dagger\sigma_z^{(1)}\mathcal{P}\,|\beta\rangle + \langle\beta|\,\mathcal{P}^\dagger\sigma_z^{(1)}\mathcal{P}\,|\alpha\rangle \tag{2}$$

Because of the limited communication used by $\mathcal{P}$ the terms in (2) are negligible, on the order of $1/\sqrt{2}^{E_{\max}-E_{\min}-4Q}$.

This means that $|\varphi\rangle$ behaves effectively like an incoherent mixture of $|\alpha\rangle$ and $|\beta\rangle$. The phase of the superposition between $|\alpha\rangle$ and $|\beta\rangle$ cannot be observed without using more communication. Similar arguments were used in [6] to argue that projecting onto $|\varphi\rangle$ required a lot of communication even given the assistance of unlimited EPR pairs. Indeed applying a phase of -1 to $|\alpha\rangle + |\beta\rangle$ but not $|\alpha\rangle - |\beta\rangle$ is equivalent to the unitary which maps $|\alpha\rangle \leftrightarrow |\beta\rangle$, and so must require quantum communication if the two states have different amounts of entanglement. (The only exception would be if an embezzling state is used.)

If $|\alpha\rangle$ and $|\beta\rangle$ are locally equivalent to maximally entangled states of different sizes then their mixture can be prepared using no communication starting with a large number of EPR pairs and a shared random bit (which can also be obtained from an EPR pair). Since $\mathcal{P}$ cannot distinguish φ from this mixture, we can run the protocol successfully starting with EPR pairs which we map for free (or almost for free, given our above use of Theorem 3) to the mixture of $|\alpha\rangle$ and $|\beta\rangle$.

To prove the full theorem we need to consider more general superpositions and new mathematical subtleties arise. But the key principle is still that a low-communication protocol cannot observe phases of superpositions between states whose degrees of entanglement are too far apart. ◀

3 The Cost of State Transformation: A Lower Bound

It is natural at this point to discuss the background and proof for Theorem 4, which establishes a lower-bound on the cost of State Transformation by the ϵ-Smoothed ℓ_∞ Earth Mover's Distance, and to postpone the discussion of Theorem 3 until Section 4, for two reasons. First, the proof of Theorem 4 in this section shares key techniques in common with the proof of Theorem 1 in Section 2 above, and so this progression may provide the reader with some continuity of thought while also reiterating the usefulness of the techniques. Second, Theorem 4 in this section motivates the notion of the ℓ_∞ Earth Mover's Distance by highlighting its, perhaps surprising, relevance to *lower* bounding the cost of state transformation. This prepares the reader with some motivation for why the *upper* bound proven in Theorem 3, in Section 4 below, is interesting and potentially useful. Thus, covering Theorem 4 at this point may provide the reader with a reason to accept the ϵ-smoothed ℓ_∞ Earth Mover's Distance as a useful proxy for the cost of State Transformation.

Whereas the proof of Theorem 3 in the next section will make direct use of Definition 2, the proof of Theorem 4 in this section is elucidated by first establishing an equivalent formulation of the ℓ_∞ Earth Mover's Distance which is derived by establishing the relationship between the ℓ_∞ Earth Mover's Distance as defined in Definition 2, and the Monge-Kantorovich Transportation distance on the real line, as shown below. After translating to this equivalent definition, stated in Definition 12, the generalization to the ϵ-smoothed ℓ_∞ Earth Mover's Distance in Definition 13 is straightforward and natural.

▶ **Definition 10.** *Given two probability distributions μ and ν on the real line, define $\Gamma(\mu,\nu)$ to be the set of probability distributions on $\mathbb{R}\times\mathbb{R}$ whose marginals are μ and ν, respectively. Given a cost function $c : \mathbb{R}\times\mathbb{R} \to [0,\infty]$ the corresponding Monge-Kantorovich distance, $d_{MK}(\mu,\nu)$ between μ and ν is defined as:*

$$d_{MK}(\mu,\nu) = \inf\left\{\int_{\mathbb{R}\times\mathbb{R}} c(x,y)d\gamma(x,y) | \gamma \in \Gamma(\mu,\nu)\right\}.$$

We can interpret $\Gamma(\mu,\nu)$ as flows mapping μ to ν and $c(x,y)$ as the cost of moving one unit of mass from position x to position y. The Monge-Kantorovich distance is then the minimum cost flow using this cost function and boundary conditions.

In order to translate into a statement about quantum states, we make the following definition in a similar style to Definition 2:

▶ **Definition 11.** *Given a bipartite shared state* $|\psi\rangle = \sum_{i\in X}\sqrt{\psi_i}\,|i\rangle\otimes|i\rangle$ *let us define a random variable* V_ψ *which takes value* $\log(\psi_i)$ *with probability* ψ_i *(note that, since the* ψ_i *sum to one, this is a well defined random variable). We now define* p_ψ *to be the probability distribution of this random variable.*

It is clear that, for every ψ, p_ψ is a probability distribution on the real line. One may note the following simple relationship between Monge-Kantorovich distance and ℓ_∞ Earth Mover's Distance:

For any $\theta > 0$, consider the Monge-Kantorovich distance, $d_{MK(\theta)}$ where the function $c : \mathbb{R}\times\mathbb{R} \to [0,\infty]$ is defined by $c(x,y) = 1$ if $|x-y| \geq \theta$ and $c(x,y) = 0$ if $|x-y| < \theta$. Then, for any two quantum states $|\psi\rangle$ and $|\phi\rangle$, we have that $d_\infty(|\psi\rangle, |\phi\rangle) < \theta$ if and only if $d_{MK(\theta)}(p_\psi, p_\phi) = 0$.

Given this concrete connection between ℓ_∞ Earth Mover's Distance and the Monge-Kantorovich distance, we can now make use of the following characterization of Monge-Kantorovich distance for distributions on the real line, which is well known in optimal transport theory:

▶ **Fact.** *Let* μ *and* ν *be probability distributions supported on the real line, and let* F_μ *and* F_ν *be their cumulative distribution functions, respectively. Then, for any* $c : \mathbb{R}\times\mathbb{R} \to [0,\infty]$ *:*

$$d_{MK}(\mu,\nu) \equiv \inf_{\gamma\in\Gamma(\mu,\nu)}\left\{\int_{\mathbb{R}\times\mathbb{R}} c(x,y)d\gamma(x,y)\right\} = \int_0^1 c(F_\mu^{-1}(s), F_\nu^{-1}(s))ds$$

It follows from this Fact, combined with the discussion above, that an equivalent definition of the ℓ_∞ Earth Mover's Distance is given by:

▶ **Definition 12.**

$$d_\infty(|\psi\rangle, |\phi\rangle) \equiv \max_{q\in[0,1]} |F_{p_\psi}^{-1}(q) - F_{p_\phi}^{-1}(q)|$$

A drawback of $d_\infty(|\psi\rangle, |\phi\rangle)$ is that it is not robust against tiny changes of either distribution in the total variation distance. Concretely, it is lower but not upper semi-continuous, since adding an infinitesimal amount of mass far away can cause $d_\infty(,\text{ t})$o increase by an unbounded amount. This is acceptable for an upper bound (i.e. protocols using communication scaling with $d_\infty(,\,)$) but it would be impossible to prove a lower bound (or no-go theorem) of the form of Theorem 4 if stated using that definition. For this reason, we find it convenient to introduce a "smoothed" version of the distance measure.

▶ **Definition 13.** *ϵ-Smoothed ℓ_∞-Earth Mover's Distance*

$$d_\infty^\epsilon(|\psi\rangle, |\phi\rangle) \equiv \max_{q\in[0,1]} \min_{r\in[q-\epsilon,q+\epsilon]} |F_{p_\psi}^{-1}(q) - F_{p_\phi}^{-1}(r)|$$

With this definition in place we can now state the lower bound.

▶ **Theorem** (Restatement of Theorem 4)**.** *Given any two bipartite shared states* $|\psi\rangle^{AB} = \sum_i \sqrt{\psi_i}\,|i\rangle^A\otimes|i\rangle^B$ *and* $|\phi\rangle^{AB} = \sum_i\sqrt{\phi_i}\,|i\rangle^A\otimes|i\rangle^B$*, shared between two parties A and*

B, together with a unitary $U_{\mathcal{P}}$ which can be performed on the state $|\psi\rangle^{AB}$ via a quantum communication protocol $\mathcal{P}$, that uses Q qubits of communication between A and B, we have that, for every ϵ:

$$\left|\langle\phi|^{AB} U_{\mathcal{P}} |\psi\rangle^{AB}\right| \leq 1 - \frac{1}{4}\epsilon^2 + 24 \cdot 2^{-\frac{1}{2}(d^{\epsilon}_{\infty}(|\psi\rangle,|\phi\rangle)-3Q)}$$

Intuitively, Theorem 4 states that two bipartite shared states which are far apart in the ϵ-Smoothed ℓ_∞-Earth Mover's Distance, cannot be made equal via a quantum communication protocol unless it uses at least $c \cdot d^{\epsilon}_{\infty}(|\psi\rangle, |\phi\rangle)$ qubits of communication (for a particular constant c which can be computed from the statement of Theorem 4).

Proof. Suppose that two bipartite shared states $|\psi\rangle$ and $|\phi\rangle$ have $d^{\epsilon}_{\infty}(|\psi\rangle, |\phi\rangle) = d$. By definition $\exists p \in [0,1]$ such that

$$\min_{r\in[p-\epsilon,p+\epsilon]} |F^{-1}_{p_\psi}(p) - F^{-1}_{p_\phi}(r)| = d \tag{3}$$

Suppose that $F^{-1}_{p_\psi}(p) < F^{-1}_{p_\phi}(r)$ (if the opposite is true then we simply switch the roles of ψ and ϕ and continue with the same proof). Define $x \equiv F^{-1}_{p_\psi}(p)$. Further define $|\psi\rangle_{\leq x} \equiv \sum_{\{i:|\log 1/\psi_i|\leq x\}} \sqrt{\psi_i}\,|i\rangle \otimes |i\rangle$, and $|\psi\rangle_{>x} \equiv |\psi\rangle - |\psi\rangle_{\leq x}$. Similarly define $|\phi\rangle_{\geq x+d} \equiv \sum_{\{i:|\log 1/\phi_i|\geq x+d\}} \sqrt{\phi_i}\,|i\rangle \otimes |i\rangle$, and $|\phi\rangle_{<x+d} \equiv |\phi\rangle - |\phi\rangle_{\geq x+d}$. Note that $|\psi\rangle_{\leq x}$, and $|\psi\rangle_{>x}$ are orthogonal, as are $|\phi\rangle_{<x+d}$ and $|\phi\rangle_{\geq x+d}$.

Since we have $x \equiv F^{-1}_{p_\psi}(p)$ it follows from the definitions that $||\,|\psi\rangle_{\leq x}\,||^2 = p$. Since $F_{p_\psi}(x) = p$, and $F^{-1}_{p_\psi}(p) < F^{-1}_{p_\phi}(r)$, it follows from Equation 3 that $F_{p_\phi}(x+d) \leq p - \epsilon$. Therefore, $||\,|\phi\rangle_{<x+d}\,||^2 \leq p - \epsilon$ and thus $||\,|\phi\rangle_{\geq x+d}\,||^2 = 1 - ||\,|\phi\rangle_{<x+d}\,||^2 \geq 1 - p + \epsilon$.

The main idea in the proof of this theorem is that we can now decompose $U_{\mathcal{P}}|\psi\rangle$ and $|\phi\rangle$ each into three nearly orthogonal parts as follows:

▶ **Definition 14.**

$$\begin{aligned}
&|\psi^1\rangle \equiv U_{\mathcal{P}}|\psi\rangle_{\leq x}, && |\phi^3\rangle \equiv |\phi\rangle_{\geq x+d}, && |\psi^2\rangle \equiv \left(I - |\phi^3\rangle\langle\phi^3|\right) U_{\mathcal{P}}|\psi\rangle_{>x}\\
&|\psi^3\rangle \equiv |\phi^3\rangle\langle\phi^3| U_{\mathcal{P}}|\psi\rangle_{>x}, && |\phi^1\rangle \equiv |\psi^1\rangle\langle\psi^1|\phi\rangle_{<x+d}, && |\phi^2\rangle \equiv \left(I - |\psi^1\rangle\langle\psi^1|\right)|\phi\rangle_{<x+d}
\end{aligned}$$

It follows from this definition that:

$$U_{\mathcal{P}}|\psi\rangle = |\psi^1\rangle + |\psi^2\rangle + |\psi^3\rangle \tag{4}$$

$$|\phi\rangle = |\phi^1\rangle + |\phi^2\rangle + |\phi^3\rangle \tag{5}$$

The motivation and key property of the particular decomposition described in Definition 14 is best illustrated by the discussion of Lemma 15 below and the remainder of the proof of Theorem 4, which follows that.

▶ **Lemma 15.** *For $i,j \in \{1,2,3\}$ with $i \neq j$, we have that $|\langle\phi^i|\psi^j\rangle| \leq h(Q,d)$, $|\langle\psi^i|\psi^j\rangle| \leq h(Q,d)$, and $|\langle\phi^i|\phi^j\rangle| \leq h(Q,d)$, where $h(Q,d) \equiv 4 \cdot 2^{\frac{3Q-d}{2}}$.*

The proof of Lemma 15 is given separately in the appendix. Within that proof is the key use of Lemma 8 which is the primary conceptual step in proving Theorem 4. Understanding the proof of Lemma 15 is also the best way of understanding the motivation behind Definition 14 above.

While the individual $|\psi^i\rangle$ and $|\phi^i\rangle$ are not necessarily all orthogonal we do have $|\psi^2\rangle \perp |\psi^3\rangle$ and $|\psi^1\rangle \perp |\psi^2\rangle + |\psi^3\rangle$. Likewise $|\phi^1\rangle \perp |\phi^2\rangle$ and $|\phi^3\rangle \perp |\psi^1\rangle + |\psi^2\rangle$. Together with equations (4) and (5), these imply

$$1 = \left\||\psi^1\rangle\right\|^2 + \left\||\psi^2\rangle\right\|^2 + \left\||\psi^3\rangle\right\|^2 \tag{6a}$$

$$1 = \left\||\phi^1\rangle\right\|^2 + \left\||\phi^2\rangle\right\|^2 + \left\||\phi^3\rangle\right\|^2 \tag{6b}$$

From Lemma 15 it follows that:

$$\begin{aligned}|\langle\phi|\mathcal{P}(\psi)\rangle| \equiv |\langle\phi|U_{\mathcal{P}}|\psi\rangle| &= \left|\left(\langle\phi^1| + \langle\phi^2| + \langle\phi^3|\right)\left(|\psi^1\rangle + |\psi^2\rangle + |\psi^3\rangle\right)\right| \\ &\leq \left|\langle\phi^1|\psi^1\rangle\right| + \left|\langle\phi^2|\psi^2\rangle\right| + \left|\langle\phi^3|\psi^3\rangle\right| + 6\cdot h(Q,d) \\ &\leq \||\phi^1\rangle\|\,\||\psi^1\rangle\| + \||\phi^2\rangle\|\,\||\psi^2\rangle\| + \||\phi^3\rangle\|\,\||\psi^3\rangle\| + 6\cdot h(Q,d) \end{aligned} \tag{7}$$

Now recall that

$$\begin{aligned}\||\psi^1\rangle\| &= \left\|U_{\mathcal{P}}|\psi\rangle_{\leq x}\right\| = \left\||\psi\rangle_{\leq x}\right\| = \sqrt{p} \\ \||\phi^3\rangle\| &= \left\||\phi\rangle_{\geq x+d}\right\| \geq \sqrt{1-p+\epsilon}\end{aligned}$$

We now return to Equation 7. Setting $x_i = \||\psi^i\rangle\|$ and $y_i = \||\phi^i\rangle\|$ for $i = 1,2,3$ we have

$$|\langle\phi|\mathcal{P}(\psi)\rangle| \leq x_1y_1 + x_2y_2 + x_3y_3 + 6\cdot h(Q,d) \tag{8}$$

where $x_1 = \sqrt{p}$, $y_3 \geq \sqrt{1-p+\epsilon}$ and $(x_1,x_2,x_3),(y_1,y_2,y_3)$ are unit vectors. We claim that this quantity is maximized by setting $x_2 = y_2 = 0$ and $y_3 = \sqrt{1-p+\epsilon}$. Indeed we can upper bound $\sqrt{p}y_1 + x_2y_2 \leq x_{12}y_{12}$ where $x_{12} \equiv \sqrt{x_1^2+x_2^2}$ and $y_{12} \equiv \sqrt{y_1^2+y_2^2}$. Now define $x_{12} = \cos(\alpha), x_3 = \sin(\alpha), y_{12} = \cos(\beta), y_3 = \sin(\beta)$ and we have

$$x_1y_1 + x_2y_2 + x_3y_3 \leq \cos(\alpha-\beta). \tag{9}$$

This is maximized by taking $(x_1,x_2,x_3) = (\sqrt{p},0,\sqrt{1-p})$ and $(y_1,y_2,y_3) = (\sqrt{p-\epsilon},0,\sqrt{1-p+\epsilon})$. Thus

$$|\langle\phi|\mathcal{P}(\psi)\rangle| \leq \sqrt{p-\epsilon}\sqrt{p} + \sqrt{1-p}\sqrt{1-p+\epsilon} + 6\cdot h(Q,d). \tag{10}$$

Finally we would like an upper bound independent of p. This maximization is performed in the proof of Fact 27 from Section G of the Appendix and yields the following.

$$|\langle\phi|\mathcal{P}(\psi)\rangle| \leq 1 - \frac{1}{4}\epsilon^2 + 6\cdot h(Q,d). \qquad \blacktriangleleft$$

4 The Cost of State Transformation: An Upper Bound

In this section we will give a proof of Theorem 3, which states that the quantum communication cost of converting between two bipartite entangled states is upper bounded by the ℓ_∞ Earth Mover's Distance between those states. This upper bound represents the second half of our two sided argument (employing both Theorem 3 and Theorem 4) that the ℓ_∞ Earth Mover's Distance is a simple and efficiently computable proxy for the cost of state conversion. The proof is divided into two parts which are proved separately in Lemma 18, and Lemma 19 together with Corollary 20. At a high level Lemma 18 tells us that, given bipartite states $|\chi\rangle$ and $|\upsilon\rangle$, one can map the Schmidt coefficients of $|\chi\rangle$ directly onto the Schmidt coefficients of $|\upsilon\rangle$ using a series of bipartite "flows" that have small degree (where degree is a quantity defined below). Lemma 19 and Corollary 20 then tell us that any such "flow" which has small degree, can be implemented as an actual bipartite state transformation, with correspondingly small communication required.

Here we establish Lemmas 18 and 19 which, together, prove the desired theorem. We begin with a couple definitions establishing the concept of flows, as we use it here.

▶ **Definition 16** (Right (Left) Index-1 Flow). *Fix two states* $|\chi\rangle = \sum_{i\in X}\sqrt{\chi_i}\,|i\rangle\otimes|i\rangle$ *and* $|\upsilon\rangle = \sum_{j\in Y}\sqrt{\upsilon_j}\,|j\rangle\otimes|j\rangle$*. A Right Index-1 Flow from* $|\chi\rangle$ *to* $|\upsilon\rangle$ *is a bipartite graph* $G_{X,Y}$ *with vertex set* $X\cup Y$*, and edge set* $E_{X,Y}$ *(where* X, Y *represents the bipartition of the vertices) such that:*

- *Each vertex in* $j\in Y$ *has degree 1 in* $G_{X,Y}$*.*
- *For all* $i\in X$*,* $\chi_i = \sum_{j\in Y:(i,j)\in E_{X,Y}}\upsilon_j$

If the roles of $|\chi\rangle$ *and* $|\upsilon\rangle$ *are reversed in the above, then we say that there is a Left Index-1 Flow from* $|\chi\rangle$ *to* $|\upsilon\rangle$*. Equivalently, there is a Left Index-1 Flow from* $|\upsilon\rangle$ *to* $|\chi\rangle$ *exactly when there is a a Right Index-1 Flow from* $|\chi\rangle$ *to* $|\upsilon\rangle$*.*

▶ **Definition 17** (Degree of a Right (Left) Index-1 Flow). *We define the degree of a Right (Left) Index-1 Flow from* $|\chi\rangle = \sum_{i\in X}\sqrt{\chi_i}\,|i\rangle\otimes|i\rangle$ *to* $|\upsilon\rangle = \sum_{j\in Y}\sqrt{\upsilon_j}\,|j\rangle\otimes|j\rangle$ *to be the maximum degree of any vertex in the bipartite graph* $G_{X,Y}$*.*

The following lemma, which a key step in proving Theorem 3, establishes that bipartite states which are close to each other in the ℓ_∞ Earth Mover's Distance of Definition 2, can be mapped to each other through a series of flows of bounded degree. This series of flows intuitively establishes a map for converting one bipartite state to the other using bounded quantum communication, in a manner that will be made rigorous in Lemma 19. The main step in the proof of Lemma 18 involves constructing a flow through a type of greedy algorithm whose analysis has a number of subtle cases. In order to concretely exhibit these cases the entire greedy algorithm, including every case, is written out in pseudocode in Algorithm 1.

▶ **Lemma 18.** *Given two states* $|\chi\rangle$ *and* $|\upsilon\rangle$*, there exist two "intermediate" states* $|\gamma\rangle$ *and* $|\rho\rangle$*, such that there is a Right Index-1 Flow from* $|\chi\rangle$ *to* $|\gamma\rangle$ *of degree at most* $2^{2\lceil d_\infty(|\chi\rangle,|\upsilon\rangle)\rceil+4}$*, a Left Index-1 Flow from* $|\gamma\rangle$ *to* $|\rho\rangle$ *of degree at most* $2^{\lceil d_\infty(|\chi\rangle,|\upsilon\rangle)\rceil+2}$*, and a Left Index-1 Flow from* $|\rho\rangle$ *to* $|\upsilon\rangle$ *of degree at most* $2^{\lceil d_\infty(|\chi\rangle,|\upsilon\rangle)\rceil+2}$*.*

The Proof of Lemma 18 is included in the Appendix, section D.

Lemma 18, above, shows that two bipartite entangled states can be connected to each other by a series of flows which have a degree which is bounded in terms of the ℓ_∞ Earth Mover's Distance between them. The next step is to establish that every flow can be implemented via a quantum communication protocol. Lemma 19 and Corollary 20, below, accomplish this by showing that, if two bipartite states can be connected by flows of small degree, then one state can be converted to the other (and vice versa) using a quantum communication protocol which only requires small amounts of communication.

▶ **Lemma 19.** *Given two states* $|\tau\rangle$ *and* $|\kappa\rangle$ *such that there is a Right Index-1 Flow from* $|\tau\rangle$ *to* $|\kappa\rangle$ *with degree at most* 2^Q*, there exists a quantum communication protocol* $\mathcal{P}$*, which uses* Q *qubits of communication, and converts the shared state* $|\tau\rangle$ *to the shared state* $|\kappa\rangle$*.*

The idea of the proof is that if $|\tau\rangle = \sum_i\sqrt{\tau_i}\,|i\rangle\otimes|i\rangle$ then it suffices to define separately protocols for each $|i\rangle\otimes|i\rangle$ term. These protocols simply use quantum communication to create a shared entangled state, resulting in the state $\sum_i\tau_i\,|i\rangle_A\otimes|i\rangle_B\otimes|\psi_i\rangle_{A'B'}$. Choosing the Schmidt coefficients according to the given Right Index-1 Flow yields the result. The details of this argument are in the Appendix xE.

Corollary 20 establishes the same result as Lemma 19, but in the reverse direction.

▶ **Corollary 20.** *Given two states* $|\tau\rangle$ *and* $|\kappa\rangle$ *such that there is a Left Index-1 Flow from* $|\kappa\rangle$ *to* $|\tau\rangle$ *with degree at most* 2^Q*, then, for two parties sharing entangled state* $|\kappa\rangle$*, there exists a quantum communication protocol* $\mathcal{P}$*, which uses* Q *qubits of communication, and converts the shared state* $|\kappa\rangle$ *to the shared state* $|\tau\rangle$*.*

The proof of Corollary 20 is straightforward and appears in Appendix F.

▶ **Theorem** (Restatement of Theorem 3). *Let $|\chi\rangle^{AB}$ and $|\upsilon\rangle^{AB}$ be two bipartite shared states. There is a protocol $\mathcal{M}_{\chi\to\upsilon}$ which can prepare $|\upsilon\rangle$ from $|\chi\rangle$, using only $4\lceil d_\infty(|\chi\rangle, |\upsilon\rangle)\rceil + 8$ qubits of communication.*

Proof. The proof follows by applying Lemma 18, followed by Lemma 19 and Corollary 20. ◀

References

1 Dorit Aharonov, Aram W. Harrow, Zeph Landau, Daniel Nagaj, Mario Szegedy, and Umesh Vazirani. Local Tests of Global Entanglement and a Counterexample to the Generalized Area Law. In *Foundations of Computer Science (FOCS), 2014 IEEE 55th Annual Symposium on*, pages 246–255, October 2014. doi:10.1109/FOCS.2014.34.

2 C. H. Bennett, H. J. Bernstein, S. Popescu, and B. Schumacher. Concentrating partial entanglement by local operations. *Phys. Rev. A*, 53:2046–2052, 1996. arXiv:quant-ph/9511030.

3 C. H. Bennett, I. Devetak, A. W. Harrow, P. W. Shor, and A. Winter. The Quantum Reverse Shannon Theorem and Resource Tradeoffs for Simulating Quantum Channels. *IEEE Trans. Inf. Theory*, 60(5):2926–2959, May 2014. doi:10.1109/TIT.2014.2309968.

4 S. Daftuar and P. Hayden. Quantum state transformations and the Schubert calculus. *Annals of Physics*, 315:80–122, 2005. arXiv:quant-ph/0410052.

5 A. W. Harrow. Entanglement spread and clean resource inequalities. In P. Exner, editor, *XVIth Int. Cong. on Math. Phys.*, pages 536–540. World Scientific, 2009. arXiv:0909.1557.

6 A. W. Harrow and D. W. Leung. A communication-efficient nonlocal measurement with application to communication complexity and bipartite gate capacities. *IEEE Trans. Inf. Theory*, 57(8):5504–5508, 2011. arXiv:0803.3066.

7 A. W. Harrow and H.-K. Lo. A tight lower bound on the classical communication cost of entanglement dilution. *IEEE Trans. Inf. Theory*, 50(2):319–327, 2004. arXiv:quant-ph/0204096.

8 P. Hayden and W. van Dam. Universal entanglement transformations without communication. *pra*, 67:060302(R), 2003. arXiv:quant-ph/0201041.

9 P. Hayden and A.J. Winter. On the communication cost of entanglement transformations. *Phys. Rev. A*, 67:012306, 2003. arXiv:quant-ph/0204092.

10 Rahul Jain, Jaikumar Radhakrishnan, and Pranab Sen. Optimal Direct Sum and Privacy Tradeoff Results for Quantum and Classical Communication Complexity, 2008. arXiv:0807.1267.

11 M. Junge and C. Palazuelos. Large Violation of Bell Inequalities with Low Entanglement. *Communications in Mathematical Physics*, 306(3):695–746, 2011. doi:10.1007/s00220-011-1296-8.

12 Debbie Leung, Ben Toner, and John Watrous. Coherent state exchange in multi-prover quantum interactive proof systems. *Chicago Journal of Theoretical Computer Science*, 11:1–18, 2013. arXiv:0804.4118.

13 H.-K. Lo and S. Popescu. The classical communication cost of entanglement manipulation: Is entanglement an inter-convertible resource? *Phys. Rev. Lett.*, 83:1459–1462, 1999. arXiv:quant-ph/9902045.

14 Ilan Newman. Private vs. common random bits in communication complexity. *Inf. Process. Lett.*, 39(2):67–71, 1991. doi:10.1016/0020-0190(91)90157-D.

15 M. A. Nielsen. Conditions for a Class of Entanglement Transformations. *Phys. Rev. Lett.*, 83:436–439, 1999. arXiv:quant-ph/9811053.

16 Oded Regev. Bell Violations Through Independent Bases Games. *Quantum Info. Comput.*, 12(1-2):9–20, January 2012. arXiv:1101.0576.

17 Thomas Vidick and Stephanie Wehner. More nonlocality with less entanglement. *Phys. Rev. A*, 83:052310, May 2011. doi:10.1103/PhysRevA.83.052310.

A Proof of Theorem 1

A concept which will be useful in the proof of Theorem 1 is the notion of the spread of a state:

▶ **Definition 21** (Spread). *For a finite dimensional bipartite entangled state* $|\psi\rangle^{AB} = \sum_i \sqrt{\psi_i}\,|i\rangle^A \otimes |i\rangle^B$ *let* λ_{max} *be the maximum of the Schmidt coefficients of* ψ*, and let* λ_{min} *be the minimum Schmidt coefficient. We define the spread of* $|\psi\rangle$ *to be the quantity* $\log(\lambda_{max}/\lambda_{min})$.

We note that the above definition of spread is given in the case of finite dimensional $|\psi\rangle$, which is the only case we will need. There is also an ϵ-smoothed variant of the spread of a state [9, 5], but it will not be needed for this proof. Within the proof of Theorem 1 the spread of a bipartite state will be used as a proxy for the amount of communication required to create that state from a maximally entangled state. This intuition is formalized, for example, by Theorem 3, but in this case of converting from a maximally entangled state, is also an implication of earlier works, such as [7, 9].

▶ **Theorem** (Restatement of Theorem 1). *Consider a quantum communication protocol* $\mathcal{R}$ *whose goal it is to compute a joint function* $g(x, y) \in \{0, 1\}$*. Suppose that* $\mathcal{R}$ *uses an arbitrary bipartite entangled state* $|\psi\rangle^{AB}$ *(of unbounded, but finite, dimension), as well as* Q *qubits of communication total, in either direction (for sufficiently large* $Q \geq 15$*). Then, for every* $\epsilon > 0$*, there exists a quantum communication protocol* $\mathcal{R}'$ *which simulates* $\mathcal{R}$ *with error* ϵ*, while using only a maximally entangled state as an entangled resource (rather than* $|\psi\rangle^{AB}$ *or any other state), and using* $O(Q/\epsilon + \log(1/\epsilon)/\epsilon)$ *qubits of communication. Thus, if* $\mathcal{R}$ *computes* f *with error* ϵ' *it follows that* $\mathcal{R}'$ *computes* f *with error* $\epsilon + \epsilon'$.

Proof. Given $\mathcal{R}$, g, and $|\psi\rangle$ as in the theorem statement, Schmidt decompose $|\psi\rangle$ as $\sum_i \sqrt{\lambda_i}\,|i, i\rangle$ (see Remark 6 for why we may assume WLOG that $|\psi\rangle$ has this form).

Let $N \geq 2$ be an integer, which will be specified later. Define a function $f : [0, 1] \to \{0, 1, \ldots, 2^N\}$ given by

$$f(\lambda) = 2^{\left\lceil \left\lceil \frac{\log(1/\lambda)}{N} \right\rceil N - \log(1/\lambda) \right\rceil} \in \{1, 2, 4, \ldots, 2^N\},$$

and define a new state $|\varphi\rangle \equiv \sum_i \sum_{j \in \{1,\ldots,f(\lambda_i)\}} \sqrt{\nu_{i,j}}\,|(i, j), (i, j)\rangle$, where $\nu_{i,j} \equiv \frac{\lambda_i}{f(\lambda_i)}$. Note that $\sum_{i,j} \nu_{i,j} = 1$, so that $|\varphi\rangle$ is a normalized pure state. Furthermore, every Schmidt coefficient $\nu_{i,j}$ of $|\varphi\rangle$ is within a multiple of 2 of the integer power $2^{-\left\lceil \frac{\log(1/\lambda_i)}{N} \right\rceil N}$. This follows because

$$\begin{aligned}
\left| \log \left(\frac{\nu_{i,j}}{2^{-\left\lceil \frac{\log(1/\lambda_i)}{N} \right\rceil N}} \right) \right| &= \left| \log(\lambda_i) - \log(f(\lambda_i)) + \left\lceil \frac{\log(1/\lambda_i)}{N} \right\rceil N \right| \\
&= \left| \log(\lambda_i) - \log\left(2^{\left\lceil \left\lceil \frac{\log(1/\lambda_i)}{N} \right\rceil N - \log(1/\lambda_i) \right\rceil} \right) + \left\lceil \frac{\log(1/\lambda_i)}{N} \right\rceil N \right| \\
&= \left| \left\lceil \frac{\log(1/\lambda_i)}{N} \right\rceil N - \log(1/\lambda_i) - \left\lceil \left\lceil \frac{\log(1/\lambda_i)}{N} \right\rceil N - \log(1/\lambda_i) \right\rceil \right| \\
&\leq 1 \qquad (11)
\end{aligned}$$

Next, we can upper bound $d_\infty(|\psi\rangle, |\varphi\rangle) \leq N$ by considering the coupling in which each $\nu_{i,j}$ is moved to λ_i. The largest distance obtained here is the maximum $\log f(\lambda_i)$ for which $\lambda_i > 0$, and this in turn is $\leq N$. Therefore, by Theorem 3, there is a protocol $\mathcal{M}$ by which Alice and Bob can prepare $|\psi\rangle$ from $|\varphi\rangle$, using $4\lceil d_\infty(|\chi\rangle, |\upsilon\rangle)\rceil + 8 \leq 4N + 8$ qubits of communication. (For this special case, of course a simpler protocol could also be used.)

Define $\mathcal{C} \equiv \mathcal{R} \circ \mathcal{M}$ to be the composed protocol in which Alice and Bob start with shared state $|\varphi\rangle$, first use protocol $\mathcal{M}$ to convert $|\varphi\rangle$ to $|\psi\rangle$, and then perform protocol $\mathcal{R}$ using shared state $|\psi\rangle$ and inputs x and y, to compute the joint function $g(x, y)$. It is evident that $\mathcal{C}$ has exactly the same success probability as $\mathcal{R}$. Since $\mathcal{M}$ uses at most $4N + 8$ qubits of communication and $\mathcal{R}$ uses Q qubits of communication, $\mathcal{C}$ can be performed with $Q + 4N + 8$ qubits of communication.

For k a nonnegative integer, define $I_k := \{i : 2^{-kN+1} \geq \lambda_i > 2^{-kN-1}\}$ and define the subnormalized state

$$|\varphi_k\rangle \equiv \sum_{i \in I_k} \sqrt{\lambda_i} \, |i, i\rangle \, . \tag{12}$$

From Equation (11) and the surrounding discussion, we have that $|\varphi\rangle = \sum_k |\varphi_k\rangle$. Furthermore, by the definition of I_k, it follows that $|\varphi_k\rangle$ has spread at most 2; note that the spread of $|\varphi_k\rangle$ does not depend on whether the state is normalized or not.

The idea of the proof is that different $|\varphi_k\rangle$ are not only orthogonal, but must remain approximately orthogonal even after a small amount of quantum communication. In particular, note that for any l, $rk_{Schmidt}(|\varphi_l\rangle) \leq 2^{lN+1} \| \, |\varphi_l\rangle \, \|^2$. Furthermore, for all l we have, by definition, that the Schmidt coefficients of $|\varphi_l\rangle$ are bounded above by 2^{-lN+1}. Therefore, if U is a unitary transform using M qubits of communication, then, it follows by Lemma 8, that $\forall j, k$,

$$\begin{aligned} |\langle \varphi_k | \, U \, |\varphi_j\rangle| &\leq 2^{\frac{3}{2}M} 2^{\min(j,k)N+1} \left\| \left| \varphi_{\min(j,k)} \right\rangle \right\|^2 \sqrt{2^{-jN+1} \cdot 2^{-kN+1}} \\ &\leq 2^{\frac{3}{2}M} 2^{-N\frac{|j-k|}{2}+2} \left\| \left| \varphi_{\min(j,k)} \right\rangle \right\|^2 \end{aligned} \tag{13}$$

To apply this to our problem, we first note that the protocol $\mathcal{C}$ depends, a priori, on the inputs x, y to the function $g(x, y)$ that we wish to compute (just like the the protocol $\mathcal{R}$). We now fix any input pair x, y and for the remainder of the proof of this theorem we will perform only transformations of the shared state which do not depend on the value of x, y. We will therefore establish that our transformation to a maximally entangled shared state does not significantly impact the success probability of the quantum communication protocol *regardless* of the value of x, y. The desired Theorem then follows.

With the input x, y now fixed, we observe that the success probability of protocol $\mathcal{C}$ (which we have already established is equal to the success probability of the original protocol $\mathcal{R}$) can be expressed WLOG by performing $\mathcal{C}$ and then computing the probability of outcomes when measuring the first qubit in the computational basis. The probability that such a measurement on protocol $\mathcal{C}$ outputs $b \in \{0, 1\}$ is

$$\Pr[b] = \langle \varphi | \, \mathcal{C}^\dagger (|b\rangle \langle b| \otimes I) \mathcal{C} \, |\varphi\rangle \, ,$$

where I acts on all qubits except for the first, which is being measured. Define $\mathcal{P} \equiv \mathcal{C}^\dagger(\sigma_z \otimes I)\mathcal{C} = \mathcal{C}^\dagger(|0\rangle \langle 0| \otimes I)\mathcal{C} - \mathcal{C}^\dagger(|1\rangle \langle 1| \otimes I)\mathcal{C}$. Then

$$\Pr[0] - \Pr[1] = \langle \varphi | \, \mathcal{P} \, |\varphi\rangle = \sum_{j,k} \langle \varphi_j | \, \mathcal{P} \, |\varphi_k\rangle \tag{14}$$

Observe, for later, that $\mathcal{P}$ is a unitary operator that can be implemented using $2Q+8N+16$ qubits of communication.

The proof will proceed as follows: In Lemma 24 we show that the density matrix $\varphi = |\varphi\rangle \langle \varphi|$ can be divided into three "pieces", one piece which has small trace norm and can therefore be omitted, one piece called φ_{far} which only has non-zero terms which are far from

the diagonal in the appropriate basis, and one piece called φ_{block} which is a block-diagonal mixed state that can be produced with small error and low communication cost from a maximally entangled state. Then, in Lemma 25, we show that the φ_{far} piece of φ has very little effect on the protocol $\mathcal{C}$. This means that φ can be replaced by φ_{block} alone while incurring very little error in the success probability of $\mathcal{C}$. Stated equivalently, via the equality in Equation 14 above, Lemma 25 shows that the quantity $|\text{Tr}(\mathcal{P}(\varphi - \varphi_{\text{block}}))|$ is small. Since we know from Lemma 24 that φ_{block} can be produced with low cost from a maximally entangled state, this leads us to the desired result.

We now establish some notation which will be useful throughout the rest of the proof:

▶ **Definition 22** (subset-matrix). *Consider operators on the Hilbert space which is the span of the $|\varphi_j\rangle$. We say that an operator M' is a* subset-matrix *of an operator M, if it is the case that for all l, k either $\langle\varphi_l| M' |\varphi_k\rangle = \langle\varphi_l| M |\varphi_k\rangle$, or $\langle\varphi_l| M' |\varphi_k\rangle = 0$.*

▶ **Definition 23** (Non-Zero Set). *For an operator θ on the Hilbert space which is the span of the $|\varphi_j\rangle$, define the non-zero set of θ to be $T_\theta = \{(l,k) : \langle\varphi_k| \theta |\varphi_l\rangle \neq 0\}$.*

▶ **Lemma 24.** *Consider the density matrix $\varphi \equiv \sum_{k,l} |\varphi_k\rangle \langle\varphi_l|$. For any $\epsilon > 0$, there exist subset-matrices, $\varphi_{block}, \varphi_{far}$, of φ, such that*

1. $\|\varphi - (\varphi_{block} + \varphi_{far})\|_1 \leq 2\epsilon$
2. $T_{\varphi_{far}} \subseteq \{(l,k) : |k-l| > B\}$, *where* $B \equiv 30 + 2\left\lceil \frac{\log(1/\epsilon)}{N} \right\rceil$.
3. *The bipartite shared state φ_{block} can be prepared starting from EPR pairs with $O(N/\epsilon + \log(1/\epsilon)/\epsilon)$ bits of communication.*

The proof of Lemma 24 is included in Section B of the Appendix.

We can now bound the difference between the protocol $\mathcal{C}$ acting on φ versus $\mathcal{C}$ acting on φ_{block}, following equation 14 as follows:

$$|(\text{Pr}_\varphi[0] - \text{Pr}_\varphi[1]) - (\text{Pr}_{\varphi_{\text{block}}}[0] - \text{Pr}_{\varphi_{\text{block}}}[1])| = |\text{Tr}(\mathcal{P}(\varphi - \varphi_{\text{block}}))|$$

Setting $N = 2Q$ and recalling from the Theorem statement that $Q \geq 15$ by assumption, it follows by Lemma 25, stated below, that:

$$|(\text{Pr}_\varphi[0] - \text{Pr}_\varphi[1]) - (\text{Pr}_{\varphi_{\text{block}}}[0] - \text{Pr}_{\varphi_{\text{block}}}[1])| = |\text{Tr}(\mathcal{P}(\varphi - \varphi_{\text{block}}))| \leq 3\epsilon \tag{15}$$

This completes the proof of the Theorem as we now describe.

We know from Lemma 24 that there is a quantum communication protocol, call it $\mathcal{K}$, which prepares the shared state φ_{block} starting from just a maximally entangled state using at most $O(N/\epsilon + \log(1/\epsilon)/\epsilon)$ bits of communication. Now define the protocol $\mathcal{R}' \equiv \mathcal{C} \circ \mathcal{K}$. Since $\mathcal{C}$ uses at most $Q + 4N + 8$ qubits of communication, and since we have chosen to set $N = 2Q$ (in the line above Equation 15), it follows that $\mathcal{R}'$ uses at most $O(N/\epsilon + \log(1/\epsilon)/\epsilon) = O(Q/\epsilon + \log(1/\epsilon)/\epsilon)$ qubits of communication. Furthermore, the success probability of $\mathcal{R}'$ with only the maximally entangled state as an entangled resource is the same, by construction, as the success probability of $\mathcal{C}$ with φ_{block} as an entangled resource, which, by Equation 15 above and the original definition $\mathcal{C} \equiv \mathcal{R} \circ \mathcal{M}$, is within 3ϵ of the success probability of the original protocol $\mathcal{R}$ from the theorem statement when using the original shared state $|\psi\rangle$ as an entangled resource. This is the desired result. ◀

▶ **Lemma 25.** *For φ_{block} as constructed in Lemma 24, and for N, Q as defined in the proof of Theorem 1 we have, $|\text{Tr}(\mathcal{P}(\varphi - \varphi_{block}))| \leq 3\epsilon$ whenever $N \geq 2Q \geq 30$.*

Proof. Following Lemma 24, we define $B \equiv 30 + 2\left\lceil \frac{\log(1/\epsilon)}{N} \right\rceil$. Now, letting φ_{block} and φ_{far} be as in Lemma 24, and recalling that $\|\varphi - (\varphi_{\text{block}} + \varphi_{\text{far}})\|_1 \leq 2\epsilon$, we have:

$$\begin{aligned}
|\text{Tr}(\mathcal{P}(\varphi - \varphi_{\text{block}}))| &\leq |\text{Tr}(\mathcal{P}((\varphi_{\text{block}} + \varphi_{\text{far}}) - \varphi_{\text{block}}))| + 2\epsilon = |\text{Tr}(\mathcal{P}\varphi_{\text{far}})| + 2\epsilon \\
&= \left| \sum_{(k,l) \in T_{\varphi_{\text{far}}}} \langle \varphi_k | \mathcal{P} | \varphi_l \rangle \right| + 2\epsilon \leq \sum_{(k,l) \in T_{\varphi_{\text{far}}}} |\langle \varphi_k | \mathcal{P} | \varphi_l \rangle| + 2\epsilon \\
&\leq \sum_{k,l:|k-l|>B} |\langle \varphi_k | \mathcal{P} | \varphi_l \rangle| + 2\epsilon
\end{aligned}$$

where the final inequality follows because $T_{\varphi_{\text{far}}} \subseteq \{(l,k) : |k-l| > B\}$ by Lemma 24. Recalling that the unitary $\mathcal{P}$ can be implemented using 2Q+8N+16 qubits of communication, and applying equation 13 then gives that:

$$\begin{aligned}
|\text{Tr}(\mathcal{P}(\varphi - \varphi_{\text{block}}))| - 2\epsilon &\leq \sum_{k,l:|k-l|>B} \min(1, 2^{3/2\cdot(2Q+8N+16)} 2^{-N\frac{|k-l|}{2}+4}) \left\| |\varphi_{\min(k,l)}\rangle \right\|^2 \\
&= 2 \sum_l \||\varphi_l\rangle\|^2 \sum_{k>l+B} \min(1, 2^{3Q+12N+24} 2^{-N\frac{|k-l|}{2}+4}) \\
&= 2 \sum_{n>B} \min(1, 2^{3Q+12N+24} 2^{-N\frac{n}{2}+4}) \\
&\leq 2 \cdot 2^{3Q+12N+24} 2^{-BN/2+4} \sum_{k=0}^{\infty} 2^{-N\frac{k}{2}} \\
&= 2 \cdot 2^{3Q+12N+24} 2^{-BN/2+4} \left(1 + \frac{2^{-\frac{N}{2}}}{1 - 2^{-\frac{N}{2}}}\right) \\
&\leq 4 \cdot 2^{3Q+12N+24} 2^{-BN/2+4}
\end{aligned}$$

So, recalling from the Lemma statement that $N \geq 2Q \geq 30$ by assumption:

$$\begin{aligned}
|\text{Tr}(\mathcal{P}(\varphi - \varphi_{\text{block}}))| - 2\epsilon &\leq 4 \cdot 2^{3Q+12N+24} 2^{-BN/2+4} \\
&\leq 4 \cdot 2^{28} 2^{14N} 2^{-15N - \left\lceil \frac{\log(1/\epsilon)}{N} \right\rceil N} \\
&\leq 2^{30} 2^{-N - \log(1/\epsilon)} \\
&\leq \epsilon
\end{aligned}$$

So,

$$|\text{Tr}(\mathcal{P}(\varphi - \varphi_{\text{block}}))| \leq 3\epsilon$$ ◀

Note, in the pre-processing step in the proof of Theorem 1, and again at a point within the proof of Lemma 24 we use our Theorem 3 in a setting where either the starting or ending state is very close to a maximally entangled state. It is helpful to observe, to avoid confusion, that in such cases Theorem 3 is not strictly necessary and could be replaced with previously known results from, for example, [7, 9]. In this manuscript we will use Theorem 3 in these cases in order to remain self-contained, and for the convenience of the reader, but we emphasize that the lines of the proof of Theorem 1 in which we use Theorem 3 could be replaced with known results.

B Proof of Lemma 24

▶ **Lemma** (Restatement of Lemma 24). *Consider the density matrix $\varphi \equiv \sum_{k,l} |\varphi_k\rangle \langle\varphi_l|$. For any $\epsilon > 0$, there exist subset-matrices, $\varphi_{block}, \varphi_{far}$, of φ, such that*

1. $\|\varphi - (\varphi_{block} + \varphi_{far})\|_1 \leq 2\epsilon$
2. $T_{\varphi_{far}} \subseteq \{(l,k) : |k-l| > B\}$, *where* $B \equiv 30 + 2\left\lceil \frac{\log(1/\epsilon)}{N} \right\rceil$.
3. *The bipartite shared state φ_{block} can be prepared starting from EPR pairs with $O(N/\epsilon + \log(1/\epsilon)/\epsilon)$ bits of communication.*

Proof. Note: The terminology used in this proof is defined in the proof of Theorem 1 preceding the use of Lemma 24 there (Appendix A).

Fixing an $\epsilon > 0$ we will now show how to "cut" $\varphi \equiv \sum_{k,l} |\varphi_k\rangle \langle\varphi_l|$ down into a mixture of states of small spread such that the cut only removes subset-matrices of the operator which are either far from the diagonal or small in the trace norm (less than 2ϵ).

Define a sequence of mutually orthogonal projectors $\{P_i\}$, where each P_i is the projection onto the span of $\{|\varphi_l\rangle\}_{2(i-1)B<l\leq 2i\cdot B}$. Let

$$M_i \equiv (P_{2i-1} + P_{2i})\varphi(P_{2i-1} + P_{2i}).$$

Now, for $k \in [1, ..., \lceil 1/\epsilon \rceil]$ define

$$S_k \equiv \sum_{i=0}^{\infty} M_{i\cdot\lceil 1/\epsilon\rceil + k}.$$

The S_k are block-diagonal subset-matrices of φ, which are disjoint in the sense that $T_{S_k} \cap T_{S_{k'}} = \emptyset$ when $k \neq k'$. Additionally, $\sum_{k=1}^{\lceil 1/\epsilon\rceil} S_k = \sum_i M_i$ is a subset-matrix of φ which contains the entire diagonal of φ. Indeed $\sum_{k=1}^{\lceil 1/\epsilon\rceil} S_k$ can be obtained from φ via the "pinching" TPCP which has Kraus operators given by the $\{P_{2i-1} + P_{2i}\}$. Thus

$$1 = \mathrm{tr} \sum_{k=1}^{\lceil 1/\epsilon\rceil} S_k.$$

Choose k' such that $\mathrm{tr}[S_{k'}] \leq 1/\lceil 1/\epsilon\rceil \leq \epsilon$. Since the S_k are all PSD we also have $\|S_{k'}\|_1 \leq \epsilon$.

Our strategy now is to use something like $\varphi - S_{k'}$ as a candidate for $\varphi_{\mathrm{block}} + \varphi_{\mathrm{far}}$ in the Lemma statement. However, subtracting all of $S_{k'}$ removes some terms close to the diagonal, which, even though it is not a large fraction of all entries in φ, would make the proof and statement of Lemma 24 somewhat awkward. So, in order to make the Lemma statement as clean as possible we will only subtract the "anti-diagonal" parts of $S_{k'}$, and leave the "diagonal" parts of $S_{k'}$ in a manner made precise below.

Define the block matrices

$$D_i \equiv P_{2i-1}\varphi P_{2i-1} + P_{2i}\varphi P_{2i} \qquad A_i \equiv P_{2i-1}\varphi P_{2i} + P_{2i}\varphi P_{2i-1}$$

D_i and A_i are, respectively, the diagonal and off-diagonal blocks of M_i.

Further define $K_{k'} \equiv \sum_{i=0}^{\infty} A_{i\cdot\lceil 1/\epsilon\rceil + k'}$. We have that $K_{k'} = S_{k'} - \sum_{i=0}^{\infty} D_{i\cdot\lceil 1/\epsilon\rceil + k'}$, and that $\|\sum_{i=0}^{\infty} D_{i\cdot\lceil 1/\epsilon\rceil + k'}\|_1 = \|S_{k'}\|_1$ since $\sum_{i=0}^{\infty} D_{i\cdot\lceil 1/\epsilon\rceil + k'}$ is a block-diagonal subset-matrix of $S_{k'}$ containing the entire diagonal of $S_{k'}$. Thus,

$$\|K_{k'}\|_1 = \|S_{k'} - \sum_{i=0}^{\infty} D_{i\cdot\lceil 1/\epsilon\rceil + k'}\|_1 \leq \|S_{k'}\|_1 + \|\sum_{i=0}^{\infty} D_{i\cdot\lceil 1/\epsilon\rceil + k'}\|_1 = 2\|S_{k'}\|_1 \leq 2\epsilon$$

We now define a "cut down" version of φ by $\tilde{\varphi} \equiv \varphi - K_{k'}$. From this definition we have:

$$\|\varphi - \tilde{\varphi}\|_1 = \|K_{k'}\|_1 \leq 2\epsilon. \tag{16}$$

Further, we define the projectors

$$\eta_j \equiv \sum_{2((j-1)\cdot\lceil 1/\epsilon\rceil + k') \leq l < 2(j\cdot\lceil 1/\epsilon\rceil + k')} P_l, \tag{17}$$

and define the block diagonal matrix φ_{block} as:

$$\varphi_{\text{block}} \equiv \sum_j Q_j \tilde{\varphi} Q_j = \sum_j Q_j(\varphi - K_{k'})Q_j = \sum_j Q_j \varphi Q_j \tag{18}$$

where the last equality follows because $\sum_j Q_j K_{k'} Q_j = 0$ because $K_{k'}$ consists only of the "anti-diagonal" components $A_{i\cdot\lceil 1/\epsilon\rceil + k'}$ which lie outside of the Q_j. Note that φ_{block} is a subset-matrix of $\tilde{\varphi}$ according to Definition 22. Now define φ_{far} by:

$$\varphi_{\text{far}} \equiv \tilde{\varphi} - \varphi_{\text{block}} \tag{19}$$

Therefore, φ_{far} is also a subset-matrix of $\tilde{\varphi}$ according to Definition 22. Furthermore, it follows immediately using Equation 16 that:

$$\|\varphi - (\varphi_{\text{far}} + \varphi_{\text{block}})\|_1 = \|\varphi - \tilde{\varphi}\|_1 \leq 2\epsilon \tag{20}$$

Second Claim: To establish the second claim in Lemma 24 we now show that $T_{\varphi_{\text{far}}} \subseteq \{(l,k) : |k-l| > B\}$ (recall that $B \equiv 30 + 2\left\lceil \frac{\log(1/\epsilon)}{N} \right\rceil$). To see this, we consider the case that $|k-l| \leq B$ and show that in this case $(l,k) \notin T_{\varphi_{\text{far}}}$. Assume WLOG that $k \geq l$. When $|k-l| \leq B$ we know that either $\exists j$ such that:

$$2B(2(j-1)\lceil 1/\epsilon\rceil + 2k' - 1) < l, k \leq 2B(2j\lceil 1/\epsilon\rceil + 2k' - 1) \tag{21}$$

or $\exists j$ such that:

$$4B(j\lceil 1/\epsilon\rceil + k') - 3B \leq l \leq 2B(2j\lceil 1/\epsilon\rceil + 2k' - 1) \leq k \leq 4B(j\lceil 1/\epsilon\rceil + k') - B \tag{22}$$

In the first case, denoted by Equation 21, we have that the coordinates (l,k) lie within the subset-matrix φ_{block} of φ, and thus that either $(l,k) \in T_{\varphi_{\text{block}}}$ or $(l,k) \notin T_\varphi$ by definition. In particular, either $(l,k) \in T_{Q_j \varphi Q_j} \subseteq T_{\varphi_{\text{block}}}$ as follows by Equation 18 and the definition of Q_j in Equation 17, or $(l,k) \notin T_\varphi$. If $(l,k) \in T_{\varphi_{\text{block}}}$ then we note that $T_{\varphi_{\text{block}}} \cap T_{\varphi_{\text{far}}} = \emptyset$ by definition (Equation 19), and this implies that $(l,k) \notin T_{\varphi_{\text{far}}}$. If $(l,k) \notin T_\varphi$, then $(l,k) \notin T_{\varphi_{\text{far}}}$ because $T_{\varphi_{\text{far}}} \subseteq T_\varphi$.

On the other hand, in the case denoted by Equation 22, we have the coordinates (l,k) lie within the subset-matrix $K_{k'}$ of φ, and thus that either $(l,k) \in T_{K_{k'}}$, or $(l,k) \notin T_\varphi$. The reason for this is that we know that, in this case, the coordinates (l,k) are within the subset-matrix $M_{j\lceil 1/\epsilon\rceil + k'}$ of φ. Furthermore, since we have already ruled out the case of Equation 21, we know that (l,k) is not in $D_{j\lceil 1/\epsilon\rceil + k'}$, the block diagonal portion of $M_{j\lceil 1/\epsilon\rceil + k'}$. Therefore, the coordinates (l,k) must lie in the block-anti-diagonal portion $A_{j\lceil 1/\epsilon\rceil + k'} = M_{j\lceil 1/\epsilon\rceil + k'} - D_{j\lceil 1/\epsilon\rceil + k'}$ (this can also be determined directly from Equation 22 itself, and the definition of $A_{j\lceil 1/\epsilon\rceil + k'}$). Since $K_{k'} \equiv \sum_{i=0}^{\infty} A_{i\lceil 1/\epsilon\rceil + k'}$ we know that the coordinates (l,k) lie within the $K_{k'}$, or more precisely, either $(l,k) \in T_{K_{k'}}$, or $(l,k) \notin T_\varphi$. Just as before, if $(l,k) \notin T_\varphi$, then $(l,k) \notin T_{\varphi_{\text{far}}} \subseteq T_\varphi$. On the other hand, in the case

that $(l,k) \in T_{K_{k'}}$ we know that $T_{K_{k'}} \cap T_{\varphi_{\text{far}}} = \emptyset$ because $T_{\varphi_{\text{far}}} \subseteq T_{\tilde{\varphi}}$ by Equation 19, and $T_{\tilde{\varphi}} \cap T_{K_{k'}} = \emptyset$ as follows from the definition $\tilde{\varphi} \equiv \varphi - K_{k'}$.

This establishes that $T_{\varphi_{\text{far}}} \subseteq \{(l,k) : |k-l| > B\}$.

Third Claim: To establish the third claim in Lemma 24, and complete the proof, we will show that φ_{block} is a mixture of states of spread at most $O(N/\epsilon + \log(1/\epsilon)/\epsilon)$, which means that φ_{block} can be produced from a shared maximally entangled state with at most $O(N/\epsilon + \log(1/\epsilon)/\epsilon)$ bits of communication.

Recalling the definition of φ_{block} in Equation 18, let us define $\rho'_j \equiv Q_j \varphi Q_j$, so that it is clear that $\varphi_{\text{block}} = \sum_j \rho'_j$. It is also clear that ρ'_j is not only PSD, but also an un-normalized pure state, because

$$\rho'_j \equiv Q_j \varphi Q_j = Q_j |\varphi\rangle \langle\varphi| Q_j.$$

From the definition of Q_j in Equation 17 we have that:

$$Q_j |\varphi\rangle = \sum_{B_s < l \leq B_b} |\varphi_l\rangle ,$$

Where the index limits are

$$B_s \equiv 2(2((j-1)\cdot \lceil 1/\epsilon \rceil + k') - 1)\cdot B$$
$$B_b \equiv 2(2(j\cdot \lceil 1/\epsilon \rceil + k') - 1)\cdot B.$$

We know from the definition in Equation 12 that the $|\varphi_l\rangle$ are orthogonal to each other, and that each $|\varphi_l\rangle$ has Schmidt coefficients bounded by $2^{-lN+1} \geq \lambda_i > 2^{-lN-1}$. Thus, it is immediate that ρ'_j has spread at most $(B_b - B_s)N + 4 = 2\lceil 1/\epsilon \rceil BN + 4 = O(N/\epsilon + \log(1/\epsilon)/\epsilon)$, where the last equality follows because $B = 30 + 2\left\lceil \frac{\log(1/\epsilon)}{N} \right\rceil$. Therefore φ_{block} is a normalized mixture of states with spread at most $O(N/\epsilon + \log(1/\epsilon)/\epsilon)$.

Consider the normalized version of ρ'_j, which is still a pure state of spread at most $O(N/\epsilon + \log(1/\epsilon)/\epsilon)$ it is clear that this state has Earthmover distance at most $O(N/\epsilon + \log(1/\epsilon)/\epsilon)$ from the nearest maximally entangled state (simply move all of the weight onto Schmidt coefficients of the size of the smallest Schmidt coefficient, which can be done by moving all the weight a distance less than or equal to the spread). It follows, by using Theorem 3 that there is a protocol which prepares the normalized version of ρ'_i from EPR pairs, with only $O(N/\epsilon + \log(1/\epsilon)/\epsilon)$ bits of communication (we note that this line of the proof could also have been established using result from [7, 9], for example). Now the state $\varphi_{\text{block}} \equiv \sum_i \rho'_i$ can be prepared by applying this same protocol in superposition over i (with the probability $\text{tr}(\rho'_i)$ assigned to each i), and then tracing out over the i register. Thus φ_{block} can be prepared starting from EPR pairs with $O(N/\epsilon + \log(1/\epsilon)/\epsilon)$ bits of communication. ◀

C Proof of Lemma 15

Proof. First note that it is immediate from the definitions that $\langle \phi^2 | \psi^1 \rangle = \langle \phi^3 | \psi^2 \rangle = 0$, so the conditions of the lemma are automatically satisfied in those cases.

To bound the remaining inner products we will first prove a bound on the inner product $|\langle \phi^3 | \psi^1 \rangle|$ and note that the remaining inner products are bounded as a consequence of this first bound. For notational convenience, while establishing the bound on $|\langle \phi^3 | \psi^1 \rangle|$, we set $|\rho\rangle \equiv |\psi\rangle_{\leq x}$, and let ρ_j be the non-zero Schmidt coefficients of $|\rho\rangle$ (which are just a renamed version of the non-zero Schmidt coefficients of $|\psi\rangle_{\leq x}$). Therefore, we know that, for all j, $1 \geq \rho_j \geq 2^{-x}$, and $|\psi\rangle_{\leq x} = |\rho\rangle = \sum_j \sqrt{\rho_j} |j\rangle \otimes |j\rangle$. The purpose of this renaming

convention is that we can now cleanly make the following definition. For integers i define $|\rho\rangle_i \equiv \sum_{\{j:i<|\log 1/\rho_j|\leq i+1\}} \sqrt{\rho_j}\,|j\rangle \otimes |j\rangle$, so that we have $|\psi\rangle_{\leq x} = |\rho\rangle = \sum_{i=-1}^{\lceil x\rceil} |\rho\rangle_i$, and $\langle\rho_k|\rho_i\rangle = 0$ whenever $k \neq i$. So,

$$\sum_{i=-1}^{\lceil x\rceil} \||\rho\rangle_i\|^2 = \||\rho\rangle\|^2 \leq 1 \tag{23}$$

By definition, for any $1 \leq i \leq \lceil x\rceil$, the Schmidt coefficients of $|\rho\rangle_i$ are upper bounded by 2^{-i}, and lower bounded by $2^{-(i+1)}$, and from the latter we have $rk_{Schmidt}(|\rho\rangle_i) \leq 2^{i+1}\||\rho\rangle\|^2$. Furthermore, the Schmidt coefficients of $|\phi_{\geq x+d}\rangle$ are upper bounded by $2^{-(x+d)}$, and thus, we have by Lemma 8 that:

$$\begin{aligned}&|\langle\phi_{\geq x+d}|\,U_{\mathcal{P}}\,|\rho\rangle_i| \leq 2^{\frac{3}{2}Q} rk_{Schmidt}(|\rho\rangle_i)\sqrt{2^{-(x+d)}2^{-i}} \leq 2^{\frac{3}{2}Q}\cdot 2^{i+1}\,\||\rho\rangle_i\|^2 \cdot \sqrt{2^{-(x+d)}2^{-i}}\\ &= 2\cdot 2^{\frac{3}{2}Q}\,\||\rho\rangle_i\|^2\sqrt{2^{i-x-d}} \leq 2\cdot 2^{\frac{3}{2}Q}\,\||\rho\rangle_i\|^2\cdot 2\cdot 2^{-d/2} = 4\cdot 2^{\frac{3Q-d}{2}}\,\||\rho\rangle_i\|^2,\end{aligned} \tag{24}$$

where the final inequality follows because $i \leq \lceil x\rceil$ by assumption. Thus,

$$\begin{aligned}&\left|\left\langle\phi^3\middle|\psi^1\right\rangle\right| = \left|\langle\phi_{\geq x+d}|\,U_{\mathcal{P}}\,|\psi\rangle_{\leq x}\right| = \left|\sum_{i=-1}^{\lceil x\rceil}\langle\phi_{\geq x+d}|\,U_{\mathcal{P}}\,|\rho\rangle_i\right| \leq \sum_{i=-1}^{\lceil x\rceil}|\langle\phi_{\geq x+d}|\,U_{\mathcal{P}}\,|\rho\rangle_i|\\ &\leq 4\cdot 2^{\frac{3Q-d}{2}}\sum_{i=-1}^{\lceil x\rceil}\||\rho\rangle_i\|^2 = 4\cdot 2^{\frac{3Q-d}{2}}\left\||\psi\rangle_{\leq x}\right\|^2 \leq 4\cdot 2^{\frac{3Q-d}{2}} = h(Q,d),\end{aligned} \tag{25}$$

where the second inequality follows by Equation 24 and the subsequent equality follows by Equation 23. Having established this upper bound on $\left|\left\langle\phi^3\middle|\psi^1\right\rangle\right|$ we now proceed with bounding the other inner products in the Lemma statement:

$$\left|\left\langle\psi^3\middle|\psi^1\right\rangle\right| = \left|\langle\psi_{>x}|\,U_{\mathcal{P}}^\dagger\left|\phi^3\right\rangle\left\langle\phi^3\middle|\psi^1\right\rangle\right| = \left|\langle\psi_{>x}|\,U_{\mathcal{P}}^\dagger\left|\phi^3\right\rangle\right|\left|\left\langle\phi^3\middle|\psi^1\right\rangle\right| \leq \left|\left\langle\phi^3\middle|\psi^1\right\rangle\right| \leq h(Q,d), \tag{26}$$

$$\begin{aligned}&\left|\left\langle\psi^2\middle|\psi^1\right\rangle\right| = \left|\langle\psi_{>x}|\,U_{\mathcal{P}}^\dagger\left(I-\left|\phi^3\right\rangle\left\langle\phi^3\right|\right)\left|\psi^1\right\rangle\right| \leq \left|\langle\psi_{>x}|\,U_{\mathcal{P}}^\dagger\left|\psi^1\right\rangle\right| + \left|\langle\psi_{>x}|\,U_{\mathcal{P}}^\dagger\left|\phi^3\right\rangle\left\langle\phi^3\middle|\psi^1\right\rangle\right|\\ &= \left|\langle\psi_{>x}|\,U_{\mathcal{P}}^\dagger U_{\mathcal{P}}\,|\psi\rangle_{\leq x}\right| + \left|\left\langle\psi^3\middle|\psi^1\right\rangle\right| = |\langle\psi_{>x}|\psi_{\leq x}\rangle| + \left|\left\langle\psi^3\middle|\psi^1\right\rangle\right| = \left|\left\langle\psi^3\middle|\psi^1\right\rangle\right| \leq h(Q,d),\end{aligned}$$

where both of the inequality steps follow by Equation 26 (the first of which also uses the triangle inequality).

$$\left|\left\langle\phi^3\middle|\phi^1\right\rangle\right| = \left|\left\langle\phi^3\middle|\psi^1\right\rangle\left\langle\psi^1\middle|\phi_{<x+d}\right\rangle\right| = \left|\left\langle\phi^3\middle|\psi^1\right\rangle\right|\left|\left\langle\psi^1\middle|\phi_{<x+d}\right\rangle\right| \leq \left|\left\langle\phi^3\middle|\psi^1\right\rangle\right| \leq h(Q,d),$$

$$\begin{aligned}&\left|\left\langle\phi^3\middle|\phi^2\right\rangle\right| = \left|\left\langle\phi^3\right|\left(I-\left|\psi^1\right\rangle\left\langle\psi^1\right|\right)|\phi\rangle_{<x+d}\right| \leq \left|\left\langle\phi^3\middle|\phi_{<x+d}\right\rangle\right| + \left|\left\langle\phi^3\middle|\psi^1\right\rangle\left\langle\psi^1\middle|\phi_{<x+d}\right\rangle\right|\\ &= |\langle\phi_{>x+d}|\phi_{<x+d}\rangle| + \left|\left\langle\phi^3\middle|\psi^1\right\rangle\right|\left|\left\langle\psi^1\middle|\phi_{<x+d}\right\rangle\right| = \left|\left\langle\phi^3\middle|\psi^1\right\rangle\right|\left|\left\langle\psi^1\middle|\phi_{<x+d}\right\rangle\right| \leq \left|\left\langle\phi^3\middle|\psi^1\right\rangle\right| \leq h(Q,d)\end{aligned}$$

Now, as noted earlier, $\left\langle\phi^2\middle|\psi^1\right\rangle = \left\langle\phi^3\middle|\psi^2\right\rangle = 0$. Continuing with the cross terms we have:

$$\left|\left\langle\phi^1\middle|\psi^2\right\rangle\right| = \left|\left\langle\psi^2\middle|\phi^1\right\rangle\right| = \left|\left\langle\psi^2\middle|\psi^1\right\rangle\left\langle\psi^1\middle|\phi_{<x+d}\right\rangle\right| = \left|\left\langle\psi^2\middle|\psi^1\right\rangle\right|\left|\left\langle\psi^1\middle|\phi_{<x+d}\right\rangle\right| \leq \left|\left\langle\psi^2\middle|\psi^1\right\rangle\right| \leq h(Q,d),$$

$$\left|\left\langle\phi^1\middle|\psi^3\right\rangle\right| = \left|\left\langle\psi^3\middle|\phi^1\right\rangle\right| = \left|\left\langle\psi^3\middle|\psi^1\right\rangle\left\langle\psi^1\middle|\phi_{<x+d}\right\rangle\right| = \left|\left\langle\psi^3\middle|\psi^1\right\rangle\right|\left|\left\langle\psi^1\middle|\phi_{<x+d}\right\rangle\right| \leq \left|\left\langle\psi^3\middle|\psi^1\right\rangle\right| \leq h(Q,d),$$

where the last inequality follows from Equation 26. And, since we already have $\left|\langle\phi^3|\psi^1\rangle\right| \leq h(Q,d)$ from Equation 25, the final inner product to bound is:

$$\begin{aligned}
&\left|\langle\phi^2|\psi^3\rangle\right| = \left|\langle\phi|_{<x+d}\left(I - |\psi^1\rangle\langle\psi^1|\right)|\psi^3\rangle\right| \\
&\leq \left|\langle\phi_{<x+d}|\psi^3\rangle\right| + \left|\langle\phi_{<x+d}|\psi^1\rangle\langle\psi^1|\psi^3\rangle\right| \\
&= \left|\langle\phi_{<x+d}|\phi^3\rangle\langle\phi^3|U_{\mathcal{P}}|\psi\rangle_{>x}\right| + \left|\langle\phi_{<x+d}|\psi^1\rangle\langle\psi^1|\psi^3\rangle\right| \\
&= \left|\langle\phi_{<x+d}|\phi^3\rangle\right|\left|\langle\phi^3|U_{\mathcal{P}}|\psi\rangle_{>x}\right| + \left|\langle\phi_{<x+d}|\psi^1\rangle\right|\left|\langle\psi^1|\psi^3\rangle\right| \\
&= \left|\langle\phi_{<x+d}|\phi_{>x+d}\rangle\right|\left|\langle\phi^3|U_{\mathcal{P}}|\psi\rangle_{>x}\right| + \left|\langle\phi_{<x+d}|\psi^1\rangle\right|\left|\langle\psi^1|\psi^3\rangle\right| \\
&\leq 0 + \left|\langle\psi^1|\psi^3\rangle\right| \leq h(Q,d),
\end{aligned}$$

where the last inequality follows by Equation 26. ◀

D Proof of Lemma 18

Proof. Given two states $|\chi\rangle = \sum_{i\in X}\sqrt{\chi_i}\,|i\rangle\otimes|i\rangle$ and $|\upsilon\rangle = \sum_{j\in Y}\sqrt{\upsilon_j}\,|j\rangle\otimes|j\rangle$, let $\omega(i,j) : X\times Y \to \mathbb{R}_{\geq 0}$ be the joint distribution on $X\times Y$ which satisfies the ℓ_∞ Earth Mover conditions for $|\chi\rangle$ and $|\upsilon\rangle$, and acheives the optimal earth mover bound $d_\infty(|\chi\rangle, |\upsilon\rangle)$. That is, for all $i\in X$, $\sum_{j\in Y}\omega(i,j) = \chi_i$, for all $j\in Y$, $\sum_{i\in X}\omega(i,j) = \upsilon_j$, and $\omega(i,j) = 0$ whenever $|\log(\chi_i) - \log(\upsilon_j)| > d_\infty(|\chi\rangle, |\upsilon\rangle)$.

Define $|\rho\rangle \equiv \sum_{j\in Y}\sum_{k\in[2^{\lceil d_\infty(|\chi\rangle,|\upsilon\rangle)\rceil+2}]}\sqrt{\rho_{j,k}}\,|j\rangle\otimes|k\rangle\otimes|j\rangle\otimes|k\rangle$, where

$$\rho_{j,k} \equiv \upsilon_j / 2^{\lceil d_\infty(|\chi\rangle,|\upsilon\rangle)\rceil+2}.$$

We now define the intermediate state

$$|\gamma\rangle \equiv \sum_{j\in Y}\sum_{k\in[2^{\lceil d_\infty(|\chi\rangle,|\upsilon\rangle)\rceil+2}]}\sum_{r\in[2^{\lceil d_\infty(|\chi\rangle,|\upsilon\rangle)\rceil+2}]}\sqrt{\gamma_{j,k,r}}\,|j\rangle\otimes|k\rangle\otimes|r\rangle\otimes|j\rangle\otimes|k\rangle\,|r\rangle\,,$$

where the Schmidt coefficients $\gamma_{j,k,r}$ are left unspecified for now.

In order to specify the Schmidt coefficients of the intermediate state $|\gamma\rangle$ as well as the Right Index-1 Flow from $|\chi\rangle$ to $|\gamma\rangle$, and the Left Index-1 Flow from $|\gamma\rangle$ to $|\rho\rangle$ we will first define "bins" for the Schmidt coefficients of $|\upsilon\rangle$ as follows:

For $l\in\mathbb{N}\cup\{0\}$ let $\Upsilon_l \equiv \{j\in Y : 2^{-l} \geq \upsilon_j \geq 2^{-(l+1)}\}$, and $X_l \equiv \{i\in X : 2^{-l} \geq \chi_i \geq 2^{-(l+1)}\}$. Define $\omega(X_m,\Upsilon_l) \equiv \sum_{(i,j)\in X_m\times\Upsilon_l}\omega(i,j)$.

▶ **Fact 26.** *If $|m-l| > d_\infty(|\chi\rangle, |\upsilon\rangle) + 1$, then $\omega(X_m,\Upsilon_l) = 0$*

Proof. Given $i\in X_m$, and $j\in\Upsilon_l$ we have by definition that $2^{-l}\geq\upsilon_j\geq 2^{-(l+1)}$, and $2^{-m}\geq\chi_i\geq 2^{-(m+1)}$, and therefore that $|\log(\chi_i) - \log(\upsilon_j)| \geq |m-l| - 1 > d_\infty(|\chi\rangle, |\upsilon\rangle)$, where the last equality follows by assumption. It follows by definition of $d_\infty(|\chi\rangle, |\upsilon\rangle)$ and of ω, that $\omega(i,j) = 0$. Since this is true for all $(i,j)\in X_m\times\Upsilon_l$, the claim follows. ◀

We will now specify an iterative, "greedy" procedure to define the Schmidt coefficients $\gamma_{j,k,c}$ as a function of the $|\chi\rangle$ and $|\rho\rangle$.

For each $(m,l)\in\mathbb{N}\cup\{0\}\times\mathbb{N}\cup\{0\}$ such that $\omega(X_m,\Upsilon_l) > 0$ we first note that by Fact 26 that $|m-l| < d_\infty(|\chi\rangle, |\upsilon\rangle) + 1$. Thus, for each $(i,j)\in X_m\times\Upsilon_l$,

$$\chi_i \geq 2^{-(m+1)} \geq 2^{-l-d_\infty(|\chi\rangle,|\upsilon\rangle)-2} \geq 2^{-l}/2^{\lceil d_\infty(|\chi\rangle,|\upsilon\rangle)\rceil+2} \geq \upsilon_j/2^{\lceil d_\infty(|\chi\rangle,|\upsilon\rangle)\rceil+2} \equiv \rho_{j,k}$$

for all $k\in[2^{\lceil d_\infty(|\chi\rangle,|\upsilon\rangle)\rceil+2}]$.

Algorithm 1.

For all i set $\text{temp}_i = \chi_i$
Set $i_m = \min\{X_m\}$ for all m
for $l \in \mathbb{N} \cup \{0\}$ **do**
  Set $j := \min\{Y_l\}$;
  Set $k = 0$;
  Set overflow $= 0$
  for $m \in \mathbb{N} \cup \{0\}$ **do**
    if $\omega(X_m, \Upsilon_l) > 0$ **then**
      Set $\text{temp}_\omega = \omega(X_m, \Upsilon_l)$
      while $\text{temp}_\omega > 0$ **do**
        if $\sum_{r \leq \text{overflow}} \gamma_{j,k,r} < \rho_{j,k}$ **then**
          while $\text{temp}_\omega \geq \rho_{j,k} - \sum_{r \leq \text{overflow}} \gamma_{j,k,r}$ **do**
            if $k = 2^{\lceil d_\infty(|\chi\rangle, |\upsilon\rangle)\rceil + 2}$ **then**
              Set $j = j + 1$
              Set overflow $= 0$
              Set $k = 0$
            if $\text{temp}_{i_m} < \rho_{j,k} - \sum_{r \leq \text{overflow}} \gamma_{j,k,r}$ **then**
              Set $\gamma_{j,k,\text{overflow}+1} = \text{temp}_{i_m}$
              Set $\text{temp}_\omega = \text{temp}_\omega - \text{temp}_{i_m}$
              Set $\text{temp}_{i_m} = 0$
              Add an edge in the flow graph from i_m to $(j, k, \text{overflow} + 1)$
              Set $i_m = i_m + 1$
              Set overflow $=$ overflow $+ 1$
            if $\text{temp}_{i_m} \geq \rho_{j,k} - \sum_{r \leq \text{overflow}} \gamma_{j,k,r}$ and $\text{temp}_\omega \geq \rho_{j,k} - \sum_{r \leq \text{overflow}} \gamma_{j,k,r}$ **then**
              Set $\gamma_{j,k,\text{overflow}+1} = \rho_{j,k} - \sum_{r \leq \text{overflow}} \gamma_{j,k,r}$
              Set $\text{temp}_\omega = \text{temp}_\omega - \gamma_{j,k,\text{overflow}+1}$
              Set $\text{temp}_{i_m} = \text{temp}_{i_m} - \gamma_{j,k,\text{overflow}+1}$
              Add an edge in the flow graph $G_{X,Z}$ from i_m to $(j, k, \text{overflow} + 1)$
              Set $k = k + 1$
              Set overflow $= 0$
          if $k = 2^{\lceil d_\infty(|\chi\rangle, |\upsilon\rangle)\rceil + 2}$ **then**
            Set $j = j + 1$
            Set overflow $= 0$
            Set $k = 0$
          if $\text{temp}_\omega < \rho_{j,k} - \sum_{r \leq \text{overflow}} \gamma_{j,k,r}$ **then**
            if $\text{temp}_{i_m} \leq \text{temp}_\omega$ **then**
              Set $\gamma_{j,k,\text{overflow}+1} = \text{temp}_{i_m}$
              Set $\text{temp}_\omega = \text{temp}_\omega - \text{temp}_{i_m}$
              Set $\text{temp}_{i_m} = 0$
              Add an edge in the flow graph $G_{X,Z}$ from i_m to $(j, k, \text{overflow} + 1)$
              Set $i_m = i_m + 1$
              Set overflow $=$ overflow $+ 1$
            if $\text{temp}_{i_m} \geq \text{temp}_\omega$ **then**
              Set $\gamma_{j,k,\text{overflow}+1} = \text{temp}_\omega$
              Set $\text{temp}_\omega = 0$
              Set $\text{temp}_{i_m} = \text{temp}_{i_m} - \text{temp}_\omega$
              Add an edge in the flow graph $G_{X,Z}$ from i_m to $(j, k, \text{overflow} + 1)$
              Set overflow $=$ overflow $+ 1$
            if $k = 2^{\lceil d_\infty(|\chi\rangle, |\upsilon\rangle)\rceil + 2}$ **then**
              Set $j = j + 1$
              Set overflow $= 0$
              Set $k = 0$

One may check that Algorithm 1 defines Schmidt coefficients $\gamma_{j,k,r}$, satisfying

$$\sum_{j\in Y}\sum_{k\in[2^{\lceil d_\infty(|\chi\rangle,|\upsilon\rangle)\rceil+2}]}\sum_{r\in[2^{\lceil d_\infty(|\chi\rangle,|\upsilon\rangle)\rceil+2}]}\gamma_{j,k,r}=\sum_{i\in X}\chi_i=1,$$

as well as a Right Index-1 Flow from $|\chi\rangle$ to $|\gamma\rangle$, with degree at most $2^{\lceil d_\infty(|\chi\rangle,|\upsilon\rangle)\rceil+2}\cdot 2^{\lceil d_\infty(|\chi\rangle,|\upsilon\rangle)\rceil+2}=2^{2\lceil d_\infty(|\chi\rangle,|\upsilon\rangle)\rceil+4}$. In particular the Right Index-1 Flow from $|\chi\rangle$ to $|\gamma\rangle$ is constructed in Algorithm 1 by iteratively adding edges to form the bipartite flow-graph $G_{X,Z}$ where $Z\equiv(Y,[2^{\lceil d_\infty(|\chi\rangle,|\upsilon\rangle)\rceil+2}],[2^{\lceil d_\infty(|\chi\rangle,|\upsilon\rangle)\rceil+2}])$. Each line in the pseudocode which reads "Add an edge in the flow graph from i_m to $(j,k,\text{overflow}+1)$", or similar, adds a single edge to the graph $G_{X,Z}$ and the union of all these edges forms the bipartite flow $G_{X,Z}$ between X and Z. Furthermore, for the $\gamma_{j,k,r}$ defined by Algorithm 1,

$$\sum_{r\in[2^{\lceil d_\infty(|\chi\rangle,|\upsilon\rangle)\rceil+2}]}\gamma_{j,k,r}=\rho_{j,k},$$

so that there is a Left Index-1 flow from $|\gamma\rangle$ to $|\rho\rangle$ defined by a bipartite graph between the Schmidt coefficients of $|\gamma\rangle$ and $|\rho\rangle$ respectively, in which, for every $(j,k,r)\in Y\times[2^{\lceil d_\infty(|\chi\rangle,|\upsilon\rangle)\rceil+2}]\times[2^{\lceil d_\infty(|\chi\rangle,|\upsilon\rangle)\rceil+2}]$, there is an edge from $\gamma_{j,k,r}$ to $\rho_{j,k}$ of weight $\gamma_{j,k,r}$. This Left Index-1 flow then clearly has degree $2^{\lceil d_\infty(|\chi\rangle,|\upsilon\rangle)\rceil+2}$.

Finally, recall that,

$$\sum_{k\in[2^{\lceil d_\infty(|\chi\rangle,|\upsilon\rangle)\rceil+2}]}\rho_{j,k}=\sum_{k\in[2^{\lceil d_\infty(|\chi\rangle,|\upsilon\rangle)\rceil+2}]}\upsilon_j/2^{\lceil d_\infty(|\chi\rangle,|\upsilon\rangle)\rceil+2}=\nu_j$$

So, by very similar reasoning, there is a Left Index-1 flow from $|\rho\rangle$ to $|\nu\rangle$ with degree exactly $2^{\lceil d_\infty(|\chi\rangle,|\upsilon\rangle)\rceil+2}$. ◀

E Proof of Lemma 19

Proof. By assumption there is a Right Index-1 Flow from $|\tau\rangle$ to $|\kappa\rangle$ with degree at most 2^Q, so there exists a bipartite graph $G_{X,Y}$ with vertex set $X\cup Y$, and edge set $E_{X,Y}$ (where X,Y represents the bipartition of the vertices), such that:

- Each vertex in $j\in Y$ has degree 1 in $G_{X,Y}$.
- For all $i\in X$, $\tau_i=\sum_{j\in Y:(i,j)\in E_{X,Y}}\kappa_j$.
- The maximum degree of any vertex $i\in X$ in $G_{X,Y}$ is 2^Q.

The protocol for Alice and Bob to start with shared state $|\tau\rangle$ and end up with shared state $|\kappa\rangle$ will proceed as follows: Beginning with the state $|\tau\rangle$ shared between Alice and Bob, we will refer to the register containing the Alice half of $|\tau\rangle$ as A, and the register containing the Bob half as B. Alice will append two additional registers, of Q qubits each, and initialize each of them to the all zeros state. We will call these two new registers C_1 and C_2 respectively. Alice will then perform a controlled unitary operation between A and the registers C_1 and C_2. She will then pass the register C_2 to Bob using Q qubits of quantum communication to do so. Bob will then perform a controlled unitary between B and C_2, Alice will perform a controlled unitary between A and C_1, and after that Alice and Bob will share the state $|\kappa\rangle$.

To describe the protocol more precisely we will define the specific controlled unitaries performed by Alice and Bob at each step. Beginning with a shared state $|\tau\rangle$, after Alice appends the two additional Q-qubit registers to her side of $|\tau\rangle$, the shared state looks as follows:

$$|\tau\rangle=\sum_{i\in X}\sqrt{\tau_i}\left|0^{\otimes Q}\right\rangle_{C_1}\otimes\left|0^{\otimes Q}\right\rangle_{C_2}\otimes|i\rangle_A\otimes|i\rangle_B$$

Where, initially, Alice holds the registers A, C_1, and C_2. Alice now performs a controlled unitary operation, acting on registers C_1 and C_2 and controlled on register A. To describe this controlled unitary concisely we will need to imagine that there is some total order on the elements $j \in Y$ (any total order will do, one can simply imagine that the j's are indexed by bit strings which encode integers), and we will define $s_{ij} \equiv |\{j' \in Y : j' < j, \text{ and } (i,j') \in E_{X,Y}\}|$. Note that, since every $i \in X$ has degree at most 2^Q, s_{ij} is always an integer between 0 and 2^Q, so it can always be expressed in binary as a Q-bit binary number. We will take this convention in the following argument.

Now to define Alice's controlled unitary: When controlled on $|i\rangle_A$ Alice's unitary moves the state $\left|0^{\otimes Q}\right\rangle_{C_1} \otimes \left|0^{\otimes Q}\right\rangle_{C_2}$ to the state $|i\text{-controlled}\rangle_{C_1C_2} \equiv \sum_{j \in Y:(i,j)\in E_{X,Y}} \sqrt{\kappa_j/\tau_i}\,|s_{ij}\rangle_{C_1} \otimes |s_{ij}\rangle_{C_2}$. Note that since s_{ij} is always a Q-bit binary string, it can always be contained in the Q-qubit registers C_1 and C_2. Further note that, since $\tau_i = \sum_{j \in Y:(i,j)\in E_{X,Y}} \kappa_j$ by assumption, $|i\text{-controlled}\rangle_{C_1C_2}$ is a normalized pure state. Thus there exists a unitary operation that moves $\left|0^{\otimes Q}\right\rangle_{C_1} \otimes \left|0^{\otimes Q}\right\rangle_{C_2}$ to $|i\text{-controlled}\rangle_{C_1C_2}$ and Alice need only perform this specific unitary when the control register is in state $|i\rangle_A$. So, when Alice applies this controlled unitary to her registers C_1, C_2 and A (where A is the controlling register), the resulting new shared state between Alice and Bob is:

$$|\tau\rangle = \sum_{i \in X} |i\text{-controlled}\rangle_{C_1C_2} \otimes |i\rangle_A \otimes |i\rangle_B \tag{27}$$

$$= \sum_{i \in X} \sum_{j \in Y:(i,j)\in E_{X,Y}} \sqrt{\tau_i} \cdot \sqrt{\kappa_j/\tau_i}\,|s_{ij}\rangle_{C_1} \otimes |s_{ij}\rangle_{C_2} \otimes |i\rangle_A \otimes |i\rangle_B \tag{28}$$

$$= \sum_{i \in X} \sum_{j \in Y:(i,j)\in E_{X,Y}} \sqrt{\kappa_j}\,|s_{ij}\rangle_{C_1} \otimes |s_{ij}\rangle_{C_2} \otimes |i\rangle_A \otimes |i\rangle_B \tag{29}$$

At this point Alice uses Q qubits of communication to pass the Q-qubit register C_2 to Bob. The resulting shared state is:

$$\sum_{i \in X} \sum_{j \in Y:(i,j)\in E_{X,Y}} \sqrt{\kappa_j}\,|s_{ij}\rangle_{C_1} \otimes |i\rangle_A \otimes |i\rangle_B \otimes |s_{ij}\rangle_{C_2}$$

Where Alice owns registers C_1 and A, and Bob owns registers C_2 and B. Now it is not hard to see from the definition of s_{ij} and the fact that every $j \in Y$ has degree exactly 1 in the graph $G_{X,Y}$, that there is a bijection mapping each $j \in Y$ to the tuple (i, s_{ij}). Alice and Bob both know this bijection since they know the description of $G_{X,Y}$, and since bijections are invertible, Alice and Bob can now both apply a local unitary which relabels the basis element $|i\rangle \otimes |s_{ij}\rangle$ to the basis element j. The resulting shared state is:

$$\sum_{i \in X} \sum_{j \in Y:(i,j)\in E_{X,Y}} \sqrt{\kappa_j}\,|j\rangle_A \otimes |j\rangle_B = \sum_{j \in Y} \sqrt{\kappa_j}\,|j\rangle_A \otimes |j\rangle_B \equiv |\kappa\rangle$$

Where the first equality follows because each $j \in Y$ appears in the initial sum exactly once (because j has degree exactly one in $G_{X,Y}$).

This completes the protocol. ◀

F Proof of Corollary 20

Proof. By definition, if there is a Left Index-1 Flow from $|\kappa\rangle$ to $|\tau\rangle$, then there is a Right Index-1 Flow from $|\tau\rangle$ to $|\kappa\rangle$ (which is the starting assumption of Lemma 19). One can check that, in the proof Lemma 19, every operation performed by Alice and Bob was reversible.

Therefore, the proof of this corollary is simply to start at the end of the proof of Lemma 19, and "reverse" every step of the proof in order from end to beginning (including the communication step...now communication goes from Bob to Alice rather than Alice to Bob). The result is the desired quantum communication protocol, which converts the shared state $|\kappa\rangle$ to the shared state $|\tau\rangle$ using Q qubits of communication. ◀

G Fact 27

▶ **Fact 27.** *For $p \in [0,1]$ and $0 \leq \epsilon \leq p$, $\sqrt{p-\epsilon}\sqrt{p} + \sqrt{1-p}\sqrt{1-p+\epsilon} \leq 1 - \frac{1}{8}\epsilon^2$*

Proof. Define $f(x) \equiv \sqrt{p-x}\sqrt{p}+\sqrt{1-p}\sqrt{1-p+x}$. Note that $f'(x) = -\frac{\sqrt{p}}{2\sqrt{p-x}} + \frac{\sqrt{1-p}}{2\sqrt{1-p+x}}$, and $f''(x) = -1/4\left(\frac{\sqrt{p}}{(p-x)^{3/2}} + \frac{\sqrt{1-p}}{(1-p+x)^{3/2}}\right)$. So, $f(0) = 1$, $f'(0) = 0$, and

$$f''(x) = -1/4\left(\frac{\sqrt{p}}{(p-x)^{3/2}} + \frac{\sqrt{1-p}}{(1-p+x)^{3/2}}\right) \leq -1/4\frac{\sqrt{p}}{(p-x)^{3/2}} \leq -1/4\frac{1}{p} \leq -1/4$$

for all $p \in [0,1]$ and $0 \leq x \leq p$. It follows by integration that:

$$f(x) = 1 + \int_0^x \int_0^z f''(y)dydz \leq 1 + \int_0^x \int_0^z (-1/4)dydz = 1 - \frac{1}{8}x^2$$

So,

$$\sqrt{p-\epsilon}\sqrt{p} + \sqrt{1-p}\sqrt{1-p+\epsilon} = f(\epsilon) \leq 1 - \frac{1}{8}\epsilon^2.$$ ◀

Average-Case Quantum Advantage with Shallow Circuits

François Le Gall
Graduate School of Informatics, Kyoto University,
Yoshida-Honmachi, Sakyo-ku, Kyoto 606-8501, Japan
legall@i.kyoto-u.ac.jp

Abstract

Recently Bravyi, Gosset and König (Science 2018) proved an unconditional separation between the computational powers of small-depth quantum and classical circuits for a relation. In this paper we show a similar separation in the average-case setting that gives stronger evidence of the superiority of small-depth quantum computation: we construct a computational task that can be solved on all inputs by a quantum circuit of constant depth with bounded-fanin gates (a "shallow" quantum circuit) and show that any classical circuit with bounded-fanin gates solving this problem on a non-negligible fraction of the inputs must have logarithmic depth. Our results are obtained by introducing a technique to create quantum states exhibiting global quantum correlations from any graph, via a construction that we call the *extended graph*.

Similar results have been very recently (and independently) obtained by Coudron, Stark and Vidick (`arXiv:1810.04233`), and Bene Watts, Kothari, Schaeffer and Tal (STOC 2019).

2012 ACM Subject Classification Theory of computation → Quantum computation theory

Keywords and phrases Quantum computing, circuit complexity, constant-depth circuits

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.21

Related Version A full version of the paper is available at `https://arxiv.org/abs/1810.12792`.

Acknowledgements The author is very grateful to Keisuke Fujii, Tomoyuki Morimae, Harumichi Nishimura, Ansis Rosmanis and Yasuhiro Takahashi for helpful discussions. The author also thanks Jalex Stark and Thomas Vidick for comments about the manuscript. This work was partially supported by the JSPS KAKENHI grants No. 15H01677, No. 16H01705 and No. 16H05853.

1 Introduction

1.1 Background and our results

A fundamental problem in quantum complexity theory is to prove the superiority of quantum computation over classical computation. While this has been shown in constrained models of computation such as query complexity (see for instance [4] for a recent survey), in weak models of computation like finite-state automata [21], and when considering relativized complexity classes (see, e.g., [7] for the first results and [25] for the most recent breakthrough), no definite answer is known in standard computational models such as Turing machines or general circuits. Indeed, since the complexity class BQP corresponding to the problems that can be solved efficiently by a quantum computer satisfies the inclusions $\mathsf{P} \subseteq \mathsf{BQP} \subseteq \mathsf{PSPACE}$, unconditionally separating P and BQP cannot be shown without separating P and PSPACE.

A recent active research area focuses on conditionally showing the superiority of quantum computation. Under several assumptions from computational complexity such as non-collapse of the polynomial hierarchy, the superiority of quantum computation with respect to classical computation has been shown in the standard circuit model in the worst-case setting [1, 2, 3, 11, 15, 17, 16, 23, 27] and even in the average-case setting [1, 2, 3, 8, 12, 13, 17].

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka; Article No. 21; pp. 21:1–21:20
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Note that showing the superiority in the average-case setting is a much stronger evidence of the superiority of quantum computation than a proof for the worst-case setting.

A recent breakthrough by Bravyi, Gosset and König [9, 10] showed an *unconditional* separation between the computational powers of quantum and classical small-depth circuits: they constructed a computational problem that can be solved by quantum circuits of constant depth with bounded-fanin[1] gates ("shallow quantum circuits") and showed that any classical circuit with bounded fanin gates solving this problem on all inputs must have depth $\Omega(\log m)$, where m denotes the input size. Besides being the first such unconditional separation in the circuit model, this separation is also especially important since shallow quantum circuits are likely to be the easiest quantum circuits to experimentally implement, due to their robustness to noise and decoherence. (Note that separations were already known when allowing gates with unbounded fanin or fanout [18, 20, 26]. The strength of Bravyi, Gosset and König's result is that it holds for the weaker model of quantum circuits with bounded fanin and fanout.)

The original classical lower bound shown in [9] required the classical circuit to output the correct answer (with high probability) on each input, i.e., this was only a worst-case hardness result. Showing the advantages of shallow quantum circuits for a distribution (i.e., proving a corresponding average-case hardness result), which would give a significantly stronger evidence of the advantage of quantum shallow circuits, was discussed in [9, Section 5] and referred to as a "challenging open question". The recently published journal version [10] partially answers this open question: it presents an average-case lower bound showing that any classical circuit that outputs the correct answer on a constant fraction of some restricted subset of the inputs (which can be efficiently sampled) must have logarithmic depth. In other words, it shows that any sublogarithmic-depth classical algorithm will fail with some constant probability on a input chosen uniformly at random in this restricted subset.

In this work we give a stronger average-case hardness result. Our main result is the following theorem.

▶ **Theorem 1.** *There exists a relation $\mathcal{R} \subseteq \{0,1\}^M \times \{0,1\}^N$ for which the following two assertions hold.*[2]

- *There is a constant-depth quantum circuit with bounded-fanin gates (i.e., a shallow quantum circuit) that on any input $x \in \{0,1\}^M$ outputs an element in the set $\mathcal{R}(x)$ with probability 1.*
- *There is a constant $\gamma > 0$ such that any randomized circuit C with bounded-fanin gates satisfying*

$$\frac{1}{2^M} \sum_{x \in \{0,1\}^M} \Pr[C(x) \in \mathcal{R}(x)] \geq \frac{1}{\exp(\gamma\sqrt{M})}$$

has depth $\Omega(\log M)$.

Theorem 1 thus shows the existence of a computational problem that can be solved by a shallow quantum circuit on all inputs but such that any classical circuit with bounded-fanin gates solving this problem on a non-negligible fraction of the inputs must have logarithmic

[1] In this paper the term *bounded-fanin* means, as usual, that the fanin is bounded from above by a constant.

[2] As usual in computational complexity, the subset $\mathcal{R} \subseteq \{0,1\}^M \times \{0,1\}^N$ is interpreted as the following computational problem: given an input $x \in \{0,1\}^M$, output any element of the set $\{z \in \{0,1\}^N \,|\, (x,z) \in \mathcal{R}\}$. Through this paper we will use the convenient notation $\mathcal{R}(x) = \{z \in \{0,1\}^N \mid (x,z) \in \mathcal{R}\}$, for any $x \in \{0,1\}^N$.

depth. This gives an average-case result that is a strengthening of the average-case result from [10] with respect to two aspects. First, our lower bound holds for any classical circuit that solves the problem on a non-negligible fraction of the inputs (even exponentially small), and not only on a constant fraction. Second, our statement does not make any restriction on the set of inputs for which the hardness is established, i.e., it shows that any sublogarithmic-depth classical algorithm will fail with high probability on an input chosen uniformly at random in the whole set $\{0,1\}^M$.

1.2 Overview of our techniques

Main technical result. Our central technical result is the following theorem.

▶ **Theorem 2.** *There exists a relation $R \subseteq \{0,1\}^m \times \{0,1\}^n$ for which the following two assertions hold.*

- *There is a constant-depth quantum circuit with bounded-fanin gates (i.e., a shallow quantum circuit) that on any input $x \in \{0,1\}^m$ outputs a string in the set $R(x)$ with probability 1.*
- *There is a constant $\alpha > 0$ such that any randomized circuit C with bounded-fanin gates satisfying*

$$\frac{1}{2^m} \sum_{x \in \{0,1\}^m} \Pr[C(x) \in R(x)] \geq 1 - \alpha$$

has depth $\Omega(\log m)$.

Theorem 1 is obtained from Theorem 2 by amplifying the soundness using standard techniques: the relation $\mathcal{R}$ is obtained by taking the direct product of t copies of the relation R for some sufficiently large integer t (the sizes of the inputs and outputs in $\mathcal{R}$ are thus $M = mt$ and $N = nt$). We show in Section 6 how the soundness is then amplified from $1 - \alpha$ to $(1 - \alpha)^{t'}$ with $t' \approx t$ by this process and observe that $(1 - \alpha)^{t'}$ is upper bounded by $1/\exp(\gamma\sqrt{M})$ for some constant γ. Note that this approach can also be applied to amplify the soundness of the average-case result in [10], which directly gives a result similar to Theorem 1 (but for a hard distribution that is not simply the uniform distribution).

Techniques from prior works. Before presenting our techniques we first describe how the result from [9, 10] was obtained. A central technical tool is a simple but fascinating result by Barrett et al. [5] that shows that correlations arising from local entanglement cannot be simulated classically without global interaction. This result was also used recently to show a separation between quantum and classical distributed computing [22]. More precisely, [5] considers the problem of simulating the measurement outcomes that occur when measuring each qubit of a well-chosen quantum state on n qubits (the graph state associated with a cycle of length n) in either the X-basis or the Y-basis (the choice of the basis depends on input bits), and shows that creating the resulting output distribution classically requires coordinating the outcomes of qubits located at distance $\Omega(n)$ on the cycle. This result can actually easily be adapted to show that any classical circuit with one-dimensional nearest-neighbor architecture and bounded-fanin gates requires logarithmic depth to create this distribution, since otherwise distant wires cannot interact. Since a graph state over a cycle (and more generally over any constant-degree graph) can be created using a shallow quantum circuit, this already gives an unconditional separation between the computational power of quantum shallow circuits and the computational power of this restricted class of small-depth classical circuits.

The main contribution of [9, 10] is to show how to get a similar separation without restricting the topology of the classical circuit (other than its depth, naturally). A first important observation is that while interactions can now naturally occur between distant wires, any sublogarithmic-depth bounded-fanin classical circuit C cannot create interactions between all pairs of wires. Ref. [9] showed that it is then always possible to find a large subset of wires S_C that are connected as a long cycle and in which distant wires do not interact. The key idea is then to consider a computational problem (called 2D Hidden Linear Function) where the input is divided in two parts: one part specifies the basis in which the qubits of the graph state are measured and the second part the topology of the graph state. By using the second part of the input to force the graph state to use only nodes corresponding to wires in S_C, the same argument as in [5] can be again applied on the cycle defined by S_C to conclude that the sublogarithmic-depth classical circuit C cannot output a valid output with high probability.

Our approach. Let us now describe the main ideas of our approach to prove Theorem 2. Our main technical tool, described in Section 3 is a generalization of the construction from [5]: we show how to generate useful quantum correlations not only from a cycle but also from any undirected graph G. The key insight is to consider what we call the *extended graph of* G, denoted $\overline{G}$, which is obtained by adding a vertex on each edge of G. We show that when measuring the qubits of the graph state corresponding to $\overline{G}$ in either the X-basis or the Y-basis, we get probability distributions that satisfy global conditions related to properties of subgraphs (in particular paths and cycles) of $\overline{G}$. The conditions are described in Theorems 5 and 6.

In order to prove our separations we consider a $d^3 \times d^3$ square grid in which one vertex (called a control vertex) is placed at the center of each 1×1 square of the grid (and connected by 4 edges to the 4 corners of the square), and then adding one vertex on each edge. The final graph is denoted $\overline{\mathcal{G}}_d$. The construction is described in Section 4. Note that by construction $\overline{\mathcal{G}}_d$ is an extended graph. This means that the probability distributions arising when measuring the qubits of the graph state associated with this graph, which we denote $|\overline{\mathcal{G}}_d\rangle$, can be described by Theorems 5 and 6.

We can now describe the computational problem that we consider to show our separation. Let m denote the number of control vertices in $\overline{\mathcal{G}}_d$ and n denote the total number of vertices. Observe that $m = \Theta(d^6)$ and $n = \Theta(d^6)$. Given as input a string of bits $x \in \{0,1\}^m$, we consider the following process: measure each qubit of the quantum state $|\overline{\mathcal{G}}_d\rangle$ in the X-basis except the qubits corresponding to the control vertices, which are measured either in the X-basis or in the Y-basis depending on the value of x. The relation R considered to prove Theorem 2 simply asks, given $x \in \{0,1\}^m$ as input, to compute any sequence of measurement outcomes $z \in \{0,1\}^n$ that has non-zero probability of being obtained by this process. Note that this problem can be solved by shallow quantum circuits: the graph $\overline{\mathcal{G}}_d$ has constant degree and thus the graph state $|\overline{\mathcal{G}}_d\rangle$ can be constructed in constant depth.

In Section 5 we first show that for any sublogarithmic-depth bounded-fanin classical circuit C there exists a subset S_C of wires that are connected as a long cycle and in which distant wires do not interact. The proof of this claim is similar to what was done in [9, 10]. We then show that this claim, along with Theorems 5 and 6, are enough to prove that the sublogarithmic-depth classical circuit C cannot output a valid output with high probability. The key point of our argument – and the reason why our result holds for average-case hardness on the whole set $\{0,1\}^m$ of possible inputs and not only for worst-case hardness or average-case hardness on a restricted set of inputs – is that we do not need to construct the

graph state corresponding to the subgraph induced by S_C, i.e., we do not need to adapt the topology of the measured graph state to the circuit. Theorems 5 and 6 guarantee that we can instead work with the graph state $|\overline{G}_d\rangle$ corresponding to the whole graph and simply look at the relevant part of the probability distribution (the part corresponding to the wires in S_C).

Related works. A similar result has been recently (and independently) obtained by Coudron, Stark and Vidick and expanded into a framework for robust randomness expansion [14]. The proof techniques are nevertheless different: [14] constructs a problem hard for small-depth classical circuits by starting with a non-local game and showing how to plant a polynomial number of copies of the game into a graph. Our approach, on the other hand, starts with a graph and shows how to create from it a quantum state exhibiting global quantum correlations that cannot be simulated by small-depth classical circuits with bounded-fanin gates.

An even stronger result has been very recently announced: Bene Watts, Kothari, Schaeffer and Tal [6] have shown that the 2D Hidden Linear Function introduced in [9, 10] cannot be solved on a non-negligible fraction of the inputs even by small-depth classical circuits with unbounded-fanin parity gates.

2 Preliminaries

2.1 General notations and a technical lemma

Given a Boolean function $f\colon A \to \{0,1\}$ on a finite set A, we write $|f|$ the number of elements $a \in A$ such that $f(a) = 1$, i.e., $|f| = \sum_{a\in A} f(a)$. Similarly, for any finite binary string $x \in \{0,1\}^*$, we denote $|x|$ the Hamming weight of x, i.e., the number of non-zero bits of x.

All the graphs considered in this paper will be undirected. Given a graph $G = (V, E)$ and any vertex $u \in V$, we denote

$$\mathcal{N}(u) = \{v \in V \mid \{u, v\} \in E\}$$

the set of neighbors of u. Given a path p in the graph G we will often be mainly interested only in the set of vertices on the path. For a vertex $v \in V$, we will thus use the convenient notation $v \in p$ to express the fact that v is on the path p.

The notation $\oplus$ will denote the addition modulo 2 (i.e., the bit parity). We will use the following lemma, which was first implicitly mentioned in [5], and stated formally (but in a form slightly different from the form we present below) in [9, 10]. For completeness we include a proof.

▶ **Lemma 3.** *([5, 9, 10]) Consider any affine function $q\colon \{0,1\}^3 \to \{0,1\}$ and any three affine functions $q_1\colon \{0,1\}^2 \to \{0,1\}$, $q_2\colon \{0,1\}^2 \to \{0,1\}$, $q_3\colon \{0,1\}^2 \to \{0,1\}$ such that*

$$q_1(b_2, b_3) \oplus q_2(b_1, b_3) \oplus q_3(b_1, b_2) = 0 \tag{1}$$

holds for any $(b_1, b_2, b_3) \in \{0,1\}^3$. Then at least one of the four following equalities does not hold:

$$q(0,0,0) = 0, \tag{2}$$
$$q(0,1,1) \oplus q_1(1,1) = 1, \tag{3}$$
$$q(1,0,1) \oplus q_2(1,1) = 1, \tag{4}$$
$$q(1,1,0) \oplus q_3(1,1) = 1. \tag{5}$$

Proof. Consider any affine function $q\colon \{0,1\}^3 \to \{0,1\}$ and any three affine functions $q_1, q_2, q_3\colon \{0,1\}^2 \to \{0,1\}$ satisfying Condition (1) for all $(b_1, b_2, b_3) \in \{0,1\}^3$. These four functions can be written as

$$\begin{aligned} q(b_1,b_2,b_3) &= \alpha_0 \oplus \alpha_1 b_1 \oplus \alpha_2 b_2 \oplus \alpha_3 b_3, && (6)\\ q_1(b_2,b_3) &= \beta_0 \oplus \beta_2 b_2 \oplus \beta_3 b_3, && (7)\\ q_2(b_1,b_3) &= \gamma_0 \oplus \gamma_1 b_1 \oplus \beta_3 b_3, && (8)\\ q_3(b_1,b_2) &= (\beta_0 \oplus \gamma_0) \oplus \gamma_1 b_1 \oplus \beta_2 b_2. && (9) \end{aligned}$$

for some coefficients $\alpha_0, \alpha_1, \alpha_2, \alpha_3, \beta_0, \beta_2, \beta_3, \gamma_0, \gamma_1 \in \{0,1\}$. Assume that these functions satisfy all the four equations (2)-(5). Equation (2) implies that $\alpha_0 = 0$. Consider the quantity

$$\lambda = q(1,1,0) \oplus q_1(1,1) \oplus q(0,1,1) \oplus q_2(1,1) \oplus q(1,0,1) \oplus q_3(1,1).$$

Computing this quantity using the four equations (6)-(9) gives $\lambda = 3\alpha_0 = 0$. On the other hand, computing λ using the three equations (3)-(5) gives $\lambda = 1 \oplus 1 \oplus 1 = 1$, which leads to a contradiction and implies that the four equations (2)-(5) cannot hold simultaneously. ◀

2.2 Quantum computation: graph states and their measurements

Quantum gates. We assume that the reader is familiar with the basics of quantum computation and refer to [24] for a standard reference. We will use the Hadamard gate H and the Pauli X, Y and Z gates:

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix},$$

where i denotes the imaginary unit of complex numbers. Note that $XZ = -ZX = -iY$. We will use two kinds of measurements: measurements in the X-basis and measurements in the Y-basis, which correspond to projective measurements with observables X and Y, respectively. Concretely, a measurement in the X-basis is realized by applying a Hadamard gate to this qubit and then measuring it in the computational basis $\{|0\rangle, |1\rangle\}$. A measurement in the Y-basis is realized by applying the gate

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -i \\ 1 & i \end{pmatrix}$$

to this qubit and then measuring it in the computational basis.[3]

Graph states. Graph states are quantum states that can be described using graphs [19]. Let $G = (V, E)$ be any undirected graph. The graph state associated with G is the quantum state on $|V|$ qubits obtained by first constructing the state

$$\bigotimes_{u \in V} |0\rangle_{\mathsf{Q}_u},$$

[3] The outcome of a measurement in the X-basis or the Y-basis is often defined as an element in $\{-1, 1\}$, i.e., the outcome corresponds to one of two eigenvalues of the observables X and Y. In our description the measurement outcome is a bit (the two bits 0 and 1 correspond to the two eigenvalues 1 and -1, respectively), which will be more convenient to describe our results.

where each Q_u represents a 1-qubit register, then applying a Hadamard gate on each register and, finally, applying a Controlled-Z gate on $(\mathsf{Q}_u, \mathsf{Q}_v)$ for any pair $\{u, v\} \in E$. We will write $|G\rangle$ the graph state associated with G.

Graph states can equivalently be defined using the stabilizer formalism. For each vertex $u \in V$ define the operator

$$\pi_u = X_u \otimes \bigotimes_{v \in \mathcal{N}(u)} Z_v,$$

where we use X_u to denote the Pauli operator X applied to Register Q_u and use Z_v to denote the Pauli operator Z applied to Register Q_v. Observe that all these operators commute, and

$$\pi_u |G\rangle = |G\rangle$$

for each $u \in G$. The graph state $|G\rangle$ is thus the simultaneous eigenstate, associated with the eigenvalue 1, of all these operators .

Measurements of graph states. The description of graph states using the stabilizer formalism is especially convenient to derive the properties of measurements we describe below (we refer to [24] for details of the general discussion of measurements of stabilizer states and state below only the properties we will use in this paper).

Consider the graph state $|G\rangle$ of a graph $G = (V, E)$. Let $U_X, U_Y \subseteq V$ be any two disjoint subsets of vertices. Assume that we measure Register Q_u, for each vertex $u \in U_X$, in the X-basis and measure Register Q_v, for each vertex $v \in U_Y$, in the Y-basis. The observable corresponding to this measurement is

$$M = \prod_{u \in U_X} X_u \prod_{v \in U_Y} Y_v.$$

For each $u \in U_X \cup U_Y$, let $z_u \in \{0, 1\}$ denote the random variable corresponding to the measurement outcome of the measurement performed on Register Q_u. Let us denote

$$z = \bigoplus_{u \in U_X \cup U_Y} z_u$$

the random variable corresponding to the parity of all the measurement outcomes. Using the stabilizer formalism it is easy to show that the value of this random variable is as follows:

- if M can be written as $M = \prod_{u \in S} \pi_u$ for some set $S \subseteq V$ then $z = 0$ with probability 1;
- if M can be written as $M = -\prod_{u \in S} \pi_u$ for some set $S \subseteq V$ then $z = 1$ with probability 1;
- if M cannot be written as $M = \prod_{u \in S} \pi_u$ or $M = -\prod_{u \in S} \pi_u$ for some set $S \subseteq V$ then $z = 0$ with probability $1/2$ and $z = 1$ with probability $1/2$.

3 Extended Graphs and their Graph States

In this section we describe the general construction on which our results are based.

For any undirected graph $G = (V, E)$, let $\overline{G}$ denote the graph with $|V| + |E|$ vertices and $2|E|$ edges obtained from G by inserting a vertex at the middle of each edge of G. We call $\overline{G}$ the *extended graph of* G. We will write V^* the set of inserted vertices and consider $\overline{G}$ as a graph over the vertex set $V \cup V^*$. We refer to Figure 1 for an illustration.

We now define the concept of f*-covering* of a graph.

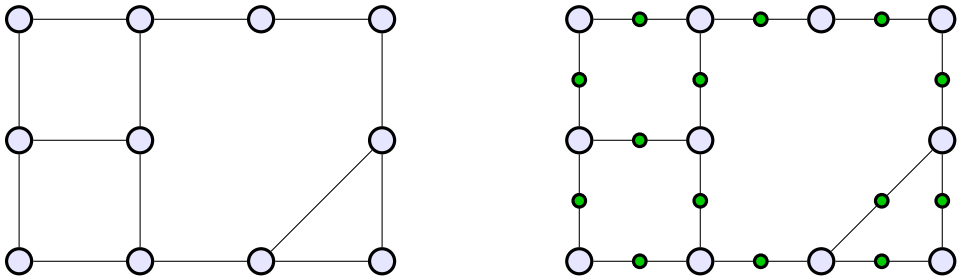


Figure 1 Example for our construction. The graph $G = (V, E)$ is represented on the left. The extended graph $\overline{G}$ is represented on the right. In this figure the large circles represent the vertices in V, while the small circles represent the vertices in V^*.

▶ **Definition 4.** *Let $G = (V, E)$ be an undirected graph and $f : V \to \{0, 1\}$ be any function such that $|f|$ is even. An f-covering of $\overline{G}$ is a set of $|f|/2$ paths of $\overline{G}$ such that each vertex in $\{v \in V \mid f(v) = 1\}$ appears once as an endpoint of one of these paths.*

We refer to Figure 2 for an illustration. Note that the $|f|/2$ paths of an f-covering do not need to be edge-disjoint.

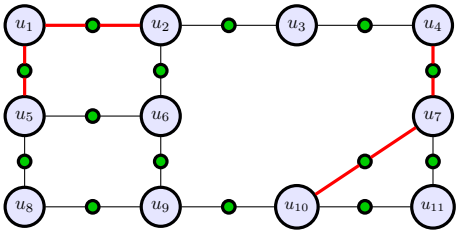


Figure 2 Illustration of the concept of f-covering. Here $V = \{u_1, \ldots, u_{11}\}$ and $f : V \to \{0, 1\}$ is defined as follows: $f(u_2) = f(u_4) = f(u_5) = f(u_{10}) = 1$ and $f(u_1) = f(u_3) = f(u_6) = f(u_7) = f(u_8) = f(u_9) = f(u_{11}) = 0$. The two paths depicted in red form an f-covering.

1. Construct the graph state over $\overline{G}$.
2. For each $v \in V$ such that $f(v) = 1$, measure the qubit of the node v in the Y-basis. Measure the qubits of all the other nodes of $\overline{G}$ in the X-basis.

Figure 3 The process $\mathcal{P}(G, f)$.

Given a graph $G = (V, E)$ and a function $f : V \to \{0, 1\}$, consider the process $\mathcal{P}(G, f)$ described in Figure 3. For any vertex $v \in V \cup V^*$, let z_v denote the random variable corresponding to the outcome of the measurement performed on the qubit of node v. The following two theorems describe the correlations among these random variables.

▶ **Theorem 5.** *For any cycle $\mathcal{C}$ of $\overline{G}$ the following equality holds with probability 1:*

$$\bigoplus_{v \in \mathcal{C} \cap V^*} z_v = 0. \tag{10}$$

Proof. In Process $\mathcal{P}(G, f)$ all the vertices in $\mathcal{C} \cap V^*$ are measured in the X-basis. Since $\mathcal{C}$ is a cycle we have

$$\prod_{v \in \mathcal{C} \cap V^*} \pi_v = \prod_{v \in \mathcal{C} \cap V^*} X_v,$$

which is the measurement operator corresponding to this measurement. The discussion of Section 2.2 implies that the parity of all the measurement outcomes is always zero, as claimed. ◀

▶ **Theorem 6.** *Assume that $|f|$ is even and let $\{p_1, \ldots, p_{|f|/2}\}$ be any f-covering of $\overline{G}$. Let us write*

$$z_V = \bigoplus_{v \in V} z_v.$$

Then the following equality holds with probability 1:

$$z_V \oplus \bigoplus_{i=1}^{|f|/2} \bigoplus_{v \in p_i \cap V^*} z_v = \begin{cases} 0 & \text{if } |f| \bmod 4 = 0, \\ 1 & \text{if } |f| \bmod 4 = 2. \end{cases} \tag{11}$$

Proof. Let $V_1 = \{u_1, \ldots, u_{|f|/2}, v_1, \ldots, v_{|f|/2}\} \subseteq V$ denote the set of vertices that appear as an endpoint of one of the paths. Let $V_2 \subseteq V^*$ denote the set of vertices in V^* that appear on an odd number of paths (remember that the paths in an f-covering do not need to be disjoint). Note that the equation we want to show (Equation (11)) can be rewritten as

$$z_V \oplus \bigoplus_{v \in V_2} z_v = \begin{cases} 0 & \text{if } |f| \bmod 4 = 0, \\ 1 & \text{if } |f| \bmod 4 = 2. \end{cases} \tag{12}$$

From the definition of an f-covering, we have $V_1 = \{v \in V \mid f(v) = 1\}$, and thus in Process $\mathcal{P}(G, f)$ all the vertices in V_1 are measured in the Y-basis, while the vertices in $V \setminus V_1$ and the vertices in V_2 are measured in the X-basis. The observable corresponding to this measurement is thus

$$\prod_{u \in V_1} Y_u \prod_{v \in (V \setminus V_1) \cup V_2} X_v. \tag{13}$$

Observe that

$$\prod_{v \in V} \pi_v = \prod_{v \in V} X_v.$$

This simple but crucial property follows from our construction: $\overline{G}$ is obtained from G by inserting a vertex on each edge of G. For each $i \in \{1, \ldots, |f|/2\}$ we also have

$$\prod_{v \in p_i \cap V^*} \pi_v = Z_{u_i} \left(\prod_{v \in p_i \cap V^*} X_v \right) Z_{v_i}.$$

Thus

$$\begin{aligned} \left(\prod_{v \in V} \pi_v \right) \times \left(\prod_{i=1}^{|f|/2} \prod_{v \in p_i \cap V^*} \pi_v \right) &= \left(\prod_{u \in V_1} X_u Z_u \right) \times \left(\prod_{v \in V \setminus V_1} X_v \right) \times \left(\prod_{v \in V_2} X_v \right) \\ &= (-1)^{|f|/2} \prod_{u \in V_1} Y_u \prod_{v \in (V \setminus V_1) \cup V_2} X_v. \end{aligned}$$

When $|f| \bmod 4 = 0$ the observable of Equation (13) can then be written as a product of generators of the graph state, and thus the parity of all the measurement outcomes is 0. When $|f| \bmod 4 = 2$ the additive inverse of this observable can be written as a product of generators of the graph states, and thus the parity of all the measurement outcomes is 1. This proves Equation (12), and thus Equation (11). ◀

Remark. The conditions of Equations (10) for all the cycles $\mathcal{C}$ of $\overline{G}$ and the condition of Equation (11) together actually completely characterize the distribution of the outcomes of $\mathcal{P}(G, f)$: the variables $\{z_v\}_{v \in V \cup V^*}$ are uniformly distributed over the set of all values satisfying all these equations. Note that when G is a connected graph then this corresponds to satisfying exactly $|E| - |V| + 2$ independent linear equations. Indeed, $|E| - (|V| - 1)$ equations suffice to guarantee that Equation (10) holds for all the cycles $\mathcal{C}$ of $\overline{G}$, as can be seen by considering a spanning tree of G: the spanning tree contains $|V| - 1$ edges and each of the remaining $|E| - (|V| - 1)$ edges gives rise to a cycle in G (and thus to a new linear equation) when added to the spanning tree. A similar characterization can be easily obtained when G is not connected as well, by considering separately each connected component.

4 Description of the Relation R

In this section we describe the computational problem we use to prove Theorem 2.

4.1 Our graph construction

For any even positive integer d, we explain how to construct two graphs $\mathcal{G}_d$ and $\overline{\mathcal{G}}_d$ that we will use to define the computational problems. The construction is illustrated in Figure 4.

The graph $\mathcal{G}_d$ is the graph with vertex set $V_d = V_d^1 \cup V_d^2$ defined as follows. We start with a $d^3 \times d^3$ square grid and denote V_d^1 the set of vertices of this grid (observe that $|V_d^1| = d^6$). This grid can be divided into d^4 contiguous square regions each of size $d \times d$. We call each region a box. In each box we place a vertex at the center of each 1×1 square and connect it to the four corners of the square. Let V_d^2 denote the set of all these new vertices. We have $|V_d^2| = d^4(d-1)^2$. It will be convenient to denote those vertices u_{ij} for $i, j \in \{1, \ldots, k\}$, where $k = d^2(d-1)$, with the index i representing the horizontal position and the index j representing the vertical position. This completes the description of $\mathcal{G}_d$.

The graph $\overline{\mathcal{G}}_d$ is obtained from $\mathcal{G}_d$ by the construction described in Section 3: one vertex is inserted on each edge of $\mathcal{G}_d$. Let V_d^* denote the introduced vertices. Note that $|V_d^*| = 2d^3(d^3-1) + 4d^4(d-1)^2$. Let us denote $\overline{V}_d = V_d^1 \cup V_d^2 \cup V_d^*$ the set of all vertices in $\overline{\mathcal{G}}_d$ and write

$$n = |\overline{V}_d| = \Theta(d^6).$$

For any vertex $u \in V_d$, let $\mathrm{Box}(u)$ denote the unique $d \times d$ box in which u is included. Finally, we denote $\partial(\overline{\mathcal{G}}_d)$ the external border of the graph, i.e., the perimeter of the whole grid.

4.2 Definition of the relation

Let d, k and n be as in Section 4.1. Given a matrix $A \in \{0,1\}^{k \times k}$, consider the process $\mathcal{P}_d(A)$ described in Figure 5.

In this process each node of $\overline{\mathcal{G}}_d$ performs a measurement and outputs one bit. We represent the whole output by a binary string of length n by fixing an arbitrary ordering of the n nodes of $\overline{\mathcal{G}}_d$. With this representation of measurement outcomes as strings, let

$$\Lambda_d(A) \subseteq \{0,1\}^n$$

denote the set of all the strings that occur with non-zero probability in Process $\mathcal{P}_d(A)$.

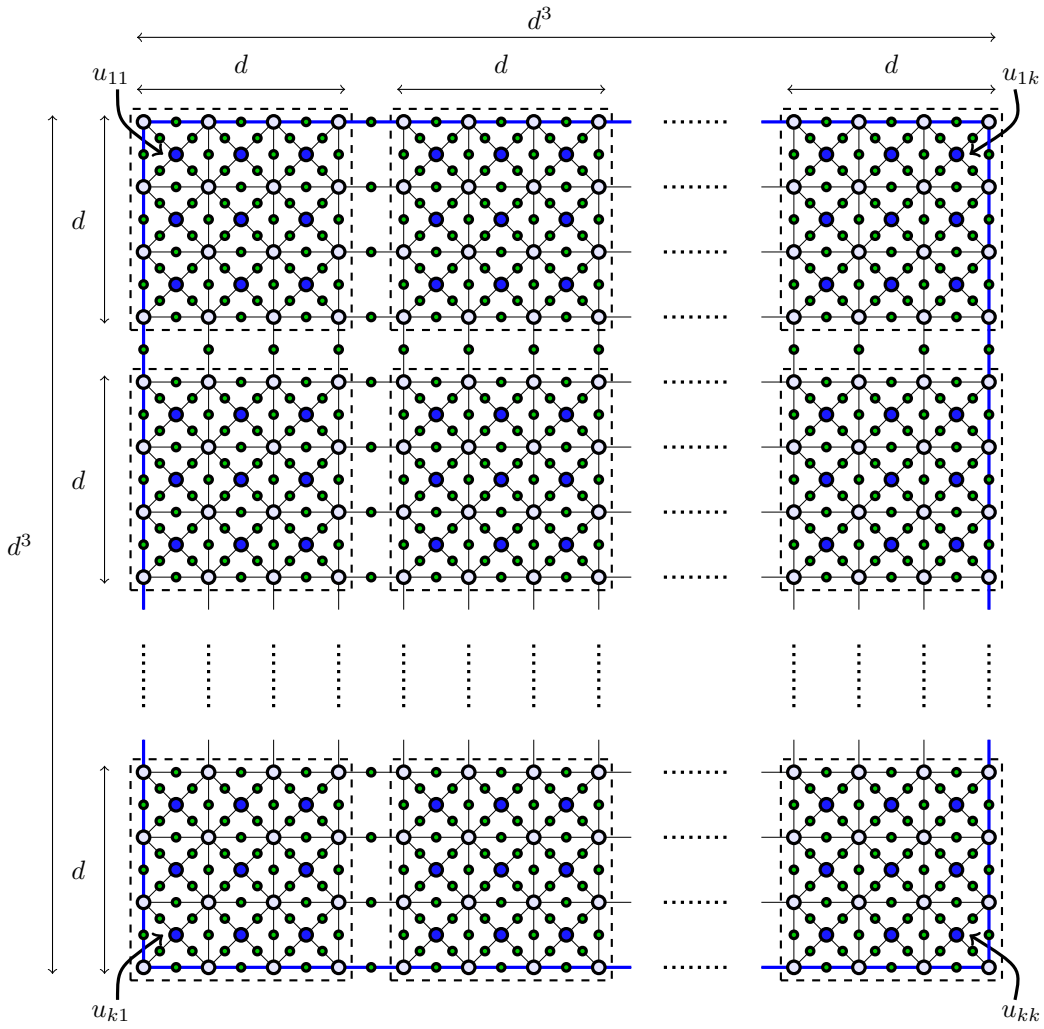


Figure 4 The graph $\overline{\mathcal{G}}_d$, here represented for $d = 4$. The vertices in V_d^1 are represented in white, the vertices in V_d^2 are represented in blue and the vertices in V_d^* are represented in green. The blue line represents the external border of the graph. The dashed squares represent the boxes.

1. Construct the graph state over $\overline{\mathcal{G}}_d$.
2. For each vertex $u_{ij} \in V_d^2$, measure the qubit of the vertex u_{ij} in the X-basis if $A_{ij} = 0$, and measure it in the Y-basis if $A_{ij} = 1$.
   For each vertex $u \in V_d^1 \cup V_d^*$, measure the qubit of the vertex u in the X-basis.

Figure 5 The process $\mathcal{P}_d(A)$.

Definition of the relation R. For any even positive integer d, the computational problem that we consider is as follows: given a matrix $A \in \{0,1\}^{k\times k}$ as input, where $k = d^2(d-1)$, compute a string from $\Lambda_d(A)$. Note that since $|\Lambda_d(A)| > 1$ there are more than one valid output. This computational problem corresponds to the relation

$$R = \big\{(A,z) \,|\, A \in \{0,1\}^{k\times k} \text{ and } z \in \Lambda_d(A)\big\} \subseteq \{0,1\}^{k\times k} \times \{0,1\}^n.$$

By setting $m = k^2$ and identifying $\{0,1\}^{k\times k}$ with $\{0,1\}^m$, we interpret R as a subset of $\{0,1\}^m \times \{0,1\}^n$. This relation R is the relation that appears in the statement of Theorem 2. To avoid confusion it will be preferable to make explicit the dependence on the parameter d. We will thus denote this relation by R_d instead of R in the next sections.

5 Proof of Theorem 2

In this section we prove Theorem 2. Let R_d be the relation defined in Section 4.2.

In the quantum setting, the computational problem corresponding to R_d can obviously be solved by directly implementing the process $\mathcal{P}_d(A)$. This can be done by a constant-depth quantum circuit since the graph $\overline{\mathcal{G}}_d$, which has constant degree, can be constructed in constant depth. Note that the description of the quantum circuit can be computed easily, e.g., by a logarithmic-space classical Turing machine.

We now show the classical lower bound, i.e., show that any classical circuit of sublogarithmic depth with bounded-fanin gates cannot output a string in $\Lambda_d(A)$ with high probability on a non-negligible fraction of the inputs A. For concreteness (and without loss of generality) we will assume in this section that all the gates in the classical circuit have fanin at most 2.

Consider any randomized classical circuit C_d, with gates of fanin at most 2, of depth at most $\frac{1}{8}\log_2 m$ for the relation R_d. The circuit has $m = k^2 = \Theta(d^6)$ input wires to receive the matrix A and n output wires. Remember that $n = \Theta(d^6)$. To simplify the presentation we assume that d is large enough so that the inequality

$$3n^{1/7} < d - 2 \tag{14}$$

holds. In Section 5.1 below we show how to associate the wires of C_d to the nodes of $\overline{\mathcal{G}}_d$. In Section 5.2 we present technical results that exploit this correspondence. Finally, in Section 5.3 we give an upper bound on the success probability of C_d and conclude the proof of Theorem 2.

5.1 Correspondence between C_d and $\overline{\mathcal{G}}_d$

We associate the wires of C_d to the nodes of $\overline{\mathcal{G}}_d$ in the following way. For any vertex $u_{ij} \in V_d^2$, we denote $x_{u_{ij}}$ the input wire of C_d that receives the entry A_{ij} of A. For any vertex $u \in \overline{V}_d$, we denote z_u the output wire of C_d that should output the outcome of the measurement performed at vertex u.

For any vertex $u \in \overline{V}_d$, we denote $L(z_u)$ the set of all vertices $v \in V_d^2$ such that the input wire x_v is in the lightcone of z_u (i.e., the value of z_u depends on the value of x_v). For any $u \in V_d^2$, we denote $L(x_u)$ the set of all vertices $v \in \overline{V}_d$ such that the output wire z_v is in the lightcone of x_u (i.e., the value of z_v depends on the value of x_u). Since the depth of C_d is at most $\frac{1}{8}\log_2 m$ and since each gate of C_d has fanin at most 2, we have $|L(z_u)| \leq m^{1/8} \leq n^{1/8}$ for each $u \in \overline{V}_d$. Let us define the set

$$\Gamma = \{u \in V_d^2 \mid L(x_u) > n^{1/7}\}.$$

Since the number of input wires is $|V_d^2| = \Theta(n)$, a simple counting argument shows that $|\Gamma| = O(n^{55/56})$, i.e., most input wires have small lightcones as well.

Define the sets $\mathcal{U}, \mathcal{V}, \mathcal{W} \subseteq V_d^2$ as follows:

$$\mathcal{U} = \left\{u_{ij} \mid i \in \{1,\dots,\lfloor k/3\rfloor\} \text{ and } j \in \{1,\dots,\lfloor k/3\rfloor\}\right\} \setminus \Gamma,$$
$$\mathcal{V} = \left\{u_{ij} \mid i \in \{\lceil 2k/3\rceil,\dots,k\} \text{ and } j \in \{1,\dots,\lfloor k/3\rfloor\}\right\} \setminus \Gamma,$$
$$\mathcal{W} = \left\{u_{ij} \mid i \in \{\lceil 2k/3\rceil,\dots,k\} \text{ and } j \in \{\lceil 2k/3\rceil,\dots,k\}\right\} \setminus \Gamma.$$

These three sets represent the vertices in $V_d^2 \cap \Gamma$ that are in the upper left part, the upper right part, and the lower right part of the graph $\overline{\mathcal{G}}_d$, respectively. From the above discussion we have $|\mathcal{U}| = \Theta(n)$, $|\mathcal{V}| = \Theta(n)$ and $|\mathcal{W}| = \Theta(n)$.

5.2 Graph-theoretic arguments

We start with a first lemma, which is similar to [9, Claim 6].

▶ **Lemma 7.** *The number of triples $(u, v, w) \in \mathcal{U} \times \mathcal{V} \times \mathcal{W}$ such that the three conditions*
- *$L(x_u) \cap \mathrm{Box}(v) = \emptyset$ and $L(x_u) \cap \mathrm{Box}(w) = \emptyset$;*
- *$L(x_v) \cap \mathrm{Box}(u) = \emptyset$ and $L(x_v) \cap \mathrm{Box}(w) = \emptyset$;*
- *$L(x_w) \cap \mathrm{Box}(u) = \emptyset$ and $L(x_w) \cap \mathrm{Box}(v) = \emptyset$.*

do not simultaneously hold is $O(n^{2+10/21})$.

Proof. Observe that for each $u \in \mathcal{U}$, there are at most $n^{1/7}$ boxes that intersect $L(x_u)$. Since each box contains $(d-1)^2 = O(n^{1/3})$ vertices in V_d^2, there are at most $O(n^{10/21})$ vertices $v \in \mathcal{V}$ such that $\mathrm{Box}(v)$ intersects $L(x_u)$. Assume that we choose a vertex v uniformly at random in $\mathcal{V}$. Then we have

$$\Pr_{v \in \mathcal{V}} [L(x_u) \cap \mathrm{Box}(v) \neq \emptyset] = O\left(n^{-11/21}\right).$$

Applying the union bound shows that if we choose a triple (u, v, w) uniformly at random in $\mathcal{U} \times \mathcal{V} \times \mathcal{W}$, then the probability that this triple does not satisfy all the three conditions of the lemma is $O(n^{-11/21})$. Since $|\mathcal{U} \times \mathcal{V} \times \mathcal{W}| = \Theta(n^3)$, we thus obtain the statement of the lemma. ◀

The following simple lemma will be crucial for our analysis.

▶ **Lemma 8.** *The number of triples $(u, v, w) \in \mathcal{U} \times \mathcal{V} \times \mathcal{W}$ such that the three lightcones $L(x_u)$, $L(x_v)$ and $L(x_w)$ are not pairwise disjoint is $O(n^{2+2/7})$.*

Proof. Let $t \in \overline{V}_d$ be any vertex of $\overline{\mathcal{G}}_d$. When choosing (u, v) uniformly at random in $\mathcal{U} \times \mathcal{V}$, the probability that t is in $L(x_u) \cap L(x_v)$ is $O((n^{1/7}/n)^2) = O(n^{-12/7})$. By the union bound this implies that when choosing a triple (u, v, w) uniformly at random in $\mathcal{U} \times \mathcal{V} \times \mathcal{W}$, the probability that x is in more than one of the three lightcones $L(x_u)$, $L(x_v)$ and $L(x_w)$ is $O(n^{-12/7})$ as well. By the union bound again, we conclude that when choosing (u, v, w) uniformly at random in $\mathcal{U} \times \mathcal{V} \times \mathcal{W}$, the probability that the three lightcones $L(x_u)$, $L(x_v)$ and $L(x_w)$ are not pairwise disjoint is $O(n^{-5/7})$. ◀

The following proposition is the main result of this subsection.

▶ **Proposition 9.** *There exists a triple of vertices $(u, v, w) \in \mathcal{U} \times \mathcal{V} \times \mathcal{W}$ such that all the following conditions hold:*

(i) *the lightcones $L(x_u)$, $L(x_v)$ and $L(x_w)$ are pairwise disjoint;*

(ii) *there exists a cycle $\mathcal{C}$ containing u, v and w such that*
- **(ii-a)** *$\mathcal{C}$ does not use any edge from the external border $\partial(\overline{\mathcal{G}}_d)$;*
- **(ii-b)** *$\mathcal{C} \cap V_d^2 = \{u, v, w\}$;*
- **(ii-c)** *$q_1 \cap L(x_w) = \emptyset$, $q_2 \cap L(x_u) = \emptyset$ and $q_3 \cap L(x_v) = \emptyset$, where q_1 denotes the direct path[4] from v to w in the cycle $\mathcal{C}$, q_2 denotes the direct path from u to w in $\mathcal{C}$ and let q_3 denote the direct path from u to v in $\mathcal{C}$.*

[4] There are two paths from v to w in the cycle $\mathcal{C}$: one path going via u and one path not using u. The direct path is the latter.

Proof. Lemmas 7 and 8 imply that among the $\Theta(n^3)$ triples $(u, v, w) \in \mathcal{U} \times \mathcal{V} \times \mathcal{W}$ there exists one triple such that Condition (i) and the three conditions of Lemma 7 simultaneously hold. Let us fix such a triple.

Let $u_1, \ldots, u_{d-2}$ denote the vertices on the right border[5] of $\mathrm{Box}(u)$ and $u'_1, \ldots, u'_{d-2}$ denote the vertices on the bottom border of $\mathrm{Box}(u)$. Similarly, let $v_1, \ldots, v_{d-2}$ denote the vertices on the left border of $\mathrm{Box}(v)$ and $v'_1, \ldots, v'_{d-2}$ denote the vertices on the bottom border of $\mathrm{Box}(v)$. Finally, let $w_1, \ldots, w_{d-2}$ denote the vertices on the top border of $\mathrm{Box}(w)$ and $w'_1, \ldots, w'_{d-2}$ denote the vertices on the left border of $\mathrm{Box}(w)$. We refer to Figure 6 for an illustration.

We can construct a path p_i^1 from u_i to v_i, a path p_i^2 from v'_i to w_i and a path p_i^3 from w'_i to u'_i, for each $i \in \{1, \ldots, d-2\}$, so that the $3(d-2)$ paths constructed are disjoint, do not use any edge on the border $\partial(\overline{\mathcal{G}}_d)$, do not go through any vertex in V_d^2, and do not contain any vertex in $\mathrm{Box}(u) \cup \mathrm{Box}(v) \cup \mathrm{Box}(w)$ except their endpoint. From Inequality (14) and since the three lightcones $L(x_u)$, $L(x_v)$ and $L(x_w)$ do not have size larger than $n^{1/7}$, there necessarily exist three indices $i_1, i_2, i_3 \in \{1, \ldots, d-2\}$ such that the three paths $p_{i_1}^1$, $p_{i_2}^2$ and $p_{i_3}^3$ do not contain any vertex in $L(x_u) \cup L(x_v) \cup L(x_v)$. Finally, observe that these three paths can be completed (avoiding all vertices in $V_d^2 \setminus \{u, v, w\}$) to obtain a cycle

$$u \to u_{i_1} \xrightarrow{p_{i_1}^1} v_{i_1} \to v \to v'_{i_2} \xrightarrow{p_{i_2}^2} w_{i_2} \to w \to w'_{i_3} \xrightarrow{p_{i_3}^3} u'_{i_3} \to u$$

that satisfies Conditions (ii-a), (ii-b) and (ii-c). See Figure 6 for an illustration. Note that Condition (ii-c) can be guaranteed due to the fact that (u, v, w) satisfies the three conditions from Lemma 7. ◀

5.3 Upper bound on the success probability

Let (u, v, w) denote the triple from $\mathcal{U} \times \mathcal{V} \times \mathcal{W}$ whose existence is guaranteed by Proposition 9. Let $\mathcal{C}$, q_1, q_2 and q_3 denote the cycle and the three paths of Condition (ii) of the proposition.

Remember that each entry A_{ij} of the input matrix A specifies the basis in which the qubit of vertex u_{ij} in the graph state $|\overline{\mathcal{G}}_d\rangle$ is measured. We will say that the vertex u_{ij} is marked if $A_{ij} = 1$. The input matrix $A \in \{0,1\}^{k \times k}$ can then be constructed by first considering the $k^2 - 3$ entries corresponding to all vertices in $V_d^2 \setminus \{u, v, w\}$, and then specifying the entries of the three vertices u, v and w. This means that A can be represented as a pair of strings (a, b) where $a \in \{0,1\}^{k^2-3}$ and $b = (b_u, b_v, b_w) \in \{0,1\}^3$.

The randomized classical circuit C_d can be seen as a deterministic circuit receiving a random string r. Let us fix the value of this random string. Let us also fix the string $a \in \{0,1\}^{k^2-3}$ and assume that the Hamming weight $|a|$ is even (note that $|a|$ corresponds to the number of marked vertices in $V_d^2 \setminus \{u, v, w\}$). The only remaining variables are thus the three bits b_u, b_v and b_w.

Observe that the graph $\overline{\mathcal{G}}_d$ remains connected when removing all the vertices on the cycle $\mathcal{C}$, due to Conditions (ii-a) and (ii-b) of Proposition 9. No vertex from $V_d^2 \setminus \{u, v, w\}$ appears in $\mathcal{C}$, from Condition (ii-b) of Proposition 9. This implies that there exists a set of $|a|/2$ paths $\{p_1, \ldots, p_{|a|/2}\}$ such that $p_i \cap \mathcal{C} = \emptyset$ for each $i \in \{1, \ldots, |a|/2\}$, and each marked vertex in $V_d^2 \setminus \{u, v, w\}$ appears once as an endpoint of one of these paths. Define

[5] To simplify the presentation we exclude the two corners at the extremities of each border.

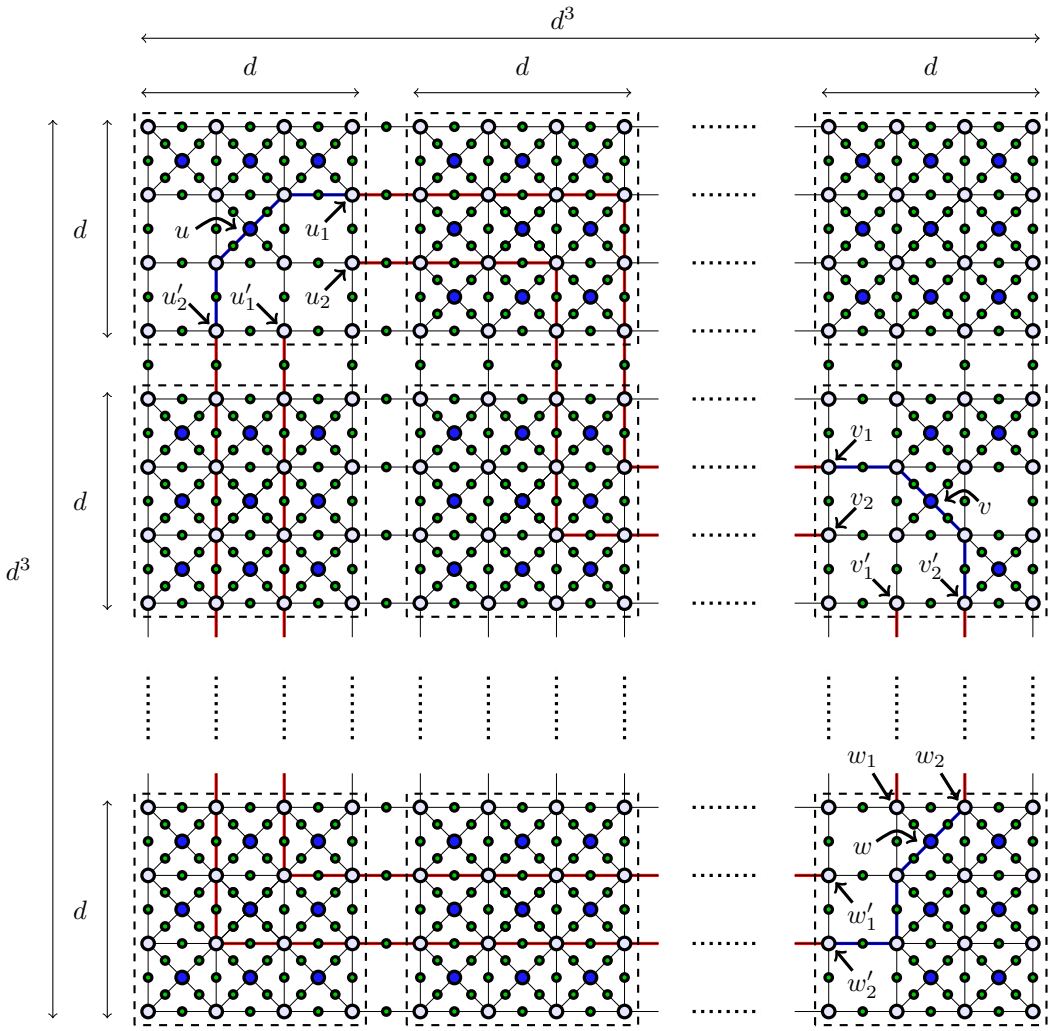


Figure 6 The paths considered to construct the cycle $\mathcal{C}$ in the proof of Proposition 9 are depicted in red. The blue line show how the paths are completed to construct the cycle $\mathcal{C}$ in the case $i_1 = 1$, $i_2 = 2$ and $i_3 = 2$. Note that some vertices in V_d^2 are omitted in order to make the figure clearer.

the three bits

$$\lambda_1 = \bigoplus_{\ell \in V_d} z_\ell,$$

$$\lambda_2 = \bigoplus_{i=1}^{|a|/2} \bigoplus_{\ell \in p_i \cap V_d^*} z_\ell,$$

$$y = \begin{cases} \lambda_1 \oplus \lambda_2 & \text{if } |a| \bmod 4 = 0, \\ \lambda_1 \oplus \lambda_2 \oplus 1 & \text{if } |a| \bmod 4 = 2. \end{cases}$$

A crucial observation is that y is an affine function of b_u, b_v and b_w, due to Condition (i) of Proposition 9.

Define

$$y_1 = \bigoplus_{\ell \in q_1 \cap V_d^*} z_\ell, \quad y_2 = \bigoplus_{\ell \in q_2 \cap V_d^*} z_\ell, \quad y_3 = \bigoplus_{\ell \in q_3 \cap V_d^*} z_\ell.$$

Condition (i) of Proposition 9 again guarantees that y_1, y_2 and y_3 are affine functions of the three bits b_u, b_v, b_w. Moreover, Condition (ii-c) implies that y_1 does not depend on b_u, y_2 does not depend on b_v and y_3 does not depend on b_w.

Theorem 5 implies that if the output of the circuit is in the set $\Lambda_d(A)$ (i.e., the output corresponds to a valid outcome arising from the corresponding measurement of the graph state $|\overline{\mathcal{G}}_d\rangle$), then the following condition should hold:

$$y_1 \oplus y_2 \oplus y_3 = 0 \quad \text{for all } (b_u, b_v, b_w) \in \{0,1\}^3. \tag{15}$$

Theorem 6 additionally implies that if the output of the circuit is in the set $\Lambda_d(A)$ then the following condition should hold:

$$\begin{cases} y = 0 & \text{if } (b_u, b_v, b_w) = (0,0,0), \\ y \oplus y_1 = 1 & \text{if } (b_u, b_v, b_w) = (0,1,1), \\ y \oplus y_2 = 1 & \text{if } (b_u, b_v, b_w) = (1,0,1), \\ y \oplus y_3 = 1 & \text{if } (b_u, b_v, b_w) = (1,1,0). \end{cases} \tag{16}$$

Lemma 7 implies that there is at least one value for the triple (b_u, b_v, b_w) for which these conditions are not satisfied.

We have just shown that for any value of r and any value of a such that $|a|$ is even, the output of the circuit C_d is incorrect for at least a fraction $1/8$ of the strings $b = (b_u, b_v, b_w) \in \{0,1\}^3$. Since $|a|$ is even with probability $1/2$ when choosing the matrix A uniformly at random, we conclude that for any value of r the output of the circuit is incorrect for at least a fraction $1/16$ of the matrices $A \in \{0,1\}^{k\times k}$. This implies the inequality

$$\sum_{A\in\{0,1\}^{k\times k}} \Pr_r[C_d(A) \notin \Lambda_d(A)] \geq \frac{2^{k^2}}{16}.$$

and thus

$$\frac{1}{2^{k^2}} \sum_{A\in\{0,1\}^{k\times k}} \Pr_r[C_d(A) \in \Lambda_d(A)] < 15/16.$$

This concludes the proof of Theorem 2.

6 Soundness Amplification for Small-Depth Circuits

In this section we show how to obtain Theorem 1 from Theorem 2. In Section 6.1 we first present a general soundness amplification result that holds for any relation. Then in Section 6.2 we apply this result to the relation R_d of Theorem 2 in order to obtain Theorem 1.

6.1 General result

Consider any relation $\mathfrak{R} \subseteq \{0,1\}^m \times \{0,1\}^n$ for some positive integers m and n. As usual, this relation is interpreted as the following computational problem: given as input a string $x \in \{0,1\}^m$, output one string from the set $\mathfrak{R}(x) = \{z \in \{0,1\}^n \mid (x,z) \in \mathfrak{R}\}$. For any integer $t \geq 1$, now consider the following computational problem: given as input t strings $x_1, \ldots, x_t \in \{0,1\}^m$, output one element from the set $\mathfrak{R}(x_1) \times \cdots \times \mathfrak{R}(x_t)$. This computational problem corresponds to the direct product of t copies of the relation $\mathfrak{R}$. We will write this relation $\mathfrak{R}^{\times t}$ and interpret it as the subset

$$\mathfrak{R}^{\times t} \subseteq \{0,1\}^{mt} \times \{0,1\}^{nt}$$

by associating $\{0,1\}^{mt}$ with the t copies of $\{0,1\}^m$ and $\{0,1\}^{nt}$ with the t copies of $\{0,1\}^n$.

The main result of this section is the following repetition theorem, which shows that if $\mathfrak{R}$ cannot be computed with average success probability larger than $1-\alpha$ using small-depth classical circuits, then $\mathfrak{R}^{\times t}$ cannot be computed with average success probability larger than $(1-\alpha)^{t'}$ for some $t' \approx t$ by circuits of the same depth. The idea is to show how to extract, from the t copies of $\mathfrak{R}$ making $\mathfrak{R}^{\times t}$, at least t' copies on which the circuit acts independently.

▶ **Theorem 10.** *Let $\mathfrak{R} \subseteq \{0,1\}^m \times \{0,1\}^n$ be a relation for which the following assertion holds for some real numbers $c \geq 0$ and $\alpha \in [0,1]$: any m-input n-output randomized circuit C with bounded-fanin gates and depth at most $c \log_2 m$ satisfies the inequality*

$$\frac{1}{2^m} \sum_{x \in \{0,1\}^m} \Pr[C(x) \in \mathfrak{R}(x)] < 1-\alpha.$$

Let t be any integer such that $t \geq 6nm^c + 2$. Then any (mt)-input (nt)-output randomized circuit C' with bounded-fanin gates and depth at most $c \log_2 m$ satisfies

$$\frac{1}{2^{mt}} \sum_{x' \in \{0,1\}^{mt}} \Pr[C'(x') \in \mathfrak{R}^{\times t}(x')] < (1-\alpha)^{t/(6m^c n+2)}.$$

Proof. Consider any (mt)-input (nt)-output randomized circuit C' with gates of fanin at most 2 and depth at most $c \log_2 m$ for the relation $\mathfrak{R}^{\times t}$. For each $i \in \{1,\ldots,t\}$, let S_i denote the set of wires corresponding to the inputs of the i-th copy of $\mathfrak{R}$ in $\mathfrak{R}^{\times t}$ and T_i denote the set of wires corresponding to the outputs of the i-th copy of $\mathfrak{R}$ in $\mathfrak{R}^{\times t}$. The following claim is the crucial part of the proof.

▷ Claim 11. There exists a subset of indices $I \subseteq \{1,\ldots,t\}$ of size $|I| \geq \frac{t}{6nm^c+2}$ such that $L(S_i) \cap T_j = \emptyset$ for all distinct $i,j \in I$.

Proof. Define the set

$$J = \Big\{ i \in \{1,\ldots,t\} \mid \sum_{x \in S_i} |L(x)| \leq 2m^c n \Big\}.$$

Since the circuit C' has depth $c \log_2 m$ and its gates have fanin at most 2, we have $|L(z)| \leq m^c$ for any output wire z. Since the total number of output wires is nt, a simple counting argument shows that $|J| \geq t/2$.

Let us now construct a graph on the vertex set J as follows: two distinct vertices $i,j \in J$ are connected by an edge if and only if at least one of $L(S_i) \cap T_j \neq \emptyset$ and $L(S_j) \cap T_i \neq \emptyset$ holds. In this graph each vertex has degree at most $2m^c n + m^c n = 3m^c n$. There thus exists[6] an independent set $I \subseteq J$ of G of size

$$|I| \geq \frac{t/2}{3m^c n+1} = \frac{t}{6m^c n+2}.$$

This independent set is precisely the set of indices we wanted to construct. ◀

To lighten the notation we will assume that the set I from Claim 11 is $I = \{1,\ldots,\ell\}$ for some integer ℓ (with $\ell \geq \frac{t}{6m^c n+2}$). This assumption can be made without loss of generality. Claim 11 implies that when the values of all the input wires in $S_{\ell+1} \cup \cdots \cup S_t$ are fixed, then

[6] Here we are using a trivial result from graph theory that states that a graph of maximum degree Δ on N vertices has an independent set of size at least $N/(\Delta+1)$.

for each $i \in \{1, \ldots, \ell\}$ the values of the output wires in T_i only depend on the values of the input wires in S_i. This implies that for any $(x_{\ell+1}, \ldots, x_t) \in \{0,1\}^{(t-\ell)m}$ the inequality

$$\frac{1}{2^{m\ell}} \sum_{x_1,\ldots,x_\ell \in \{0,1\}^m} \Pr[C'(x_1, \ldots, x_\ell, x_{\ell+1}, \ldots, x_t) \in \mathfrak{R}^{\times t}(x_1, \ldots, x_\ell, x_{\ell+1}, \ldots, x_t)] < (1-\alpha)^\ell,$$

holds, from our assumption on the relation $\mathfrak{R}$ (since the depth of C' is at most $c \log_2 m$). Thus

$$\frac{1}{2^{mt}} \sum_{x_1,\ldots,x_t \in \{0,1\}^m} \Pr[C'(x_1, \ldots, x_t) \in \mathfrak{R}^{\times t}(x_1, \ldots, x_t)] < (1-\alpha)^\ell \leq (1-\alpha)^{t/(6m^c n+2)},$$

as claimed. This concludes the proof of Theorem 10. ◀

6.2 Application: proof of Theorem 1

We are now able to give the proof of Theorem 1.

Proof of Theorem 1. We consider the relation $R_d \subseteq \{0,1\}^m \times \{0,1\}^n$ defined in Section 4.2 and used in Theorem 2. Remember that for this relation we have $m = \Theta(d^6)$ and $n = \Theta(d^6)$. Take the integer $t = \lceil (6nm^{1/8} + 2)^3 \rceil$ and observe that the inequality $t \geq m^{27/8}$ holds. Define $\mathcal{R} = R_d^{\times t}$. The sizes of the inputs and outputs in $\mathcal{R}$ are $M = mt$ and $N = nt$, respectively. Observe that $t \geq m^{27/8}$ implies $t \geq M^{27/35}$. Theorem 2 and then Theorem 10 with $\mathfrak{R} = R_d$ imply that there exist constants $c > 0$ and $\alpha > 0$ such that any M-input N-output randomized circuit C' with bounded-fanin gates and depth at most $c \log_2 m$ satisfies

$$\frac{1}{2^M} \sum_{x' \in \{0,1\}^M} \Pr[C'(x') \in \mathcal{R}(x')] < (1-\alpha)^{t^{2/3}} \leq (1-\alpha)^{M^{54/105}} < (1-\alpha)^{\sqrt{M}},$$

which leads to the claimed statement. ◀

References

1. Scott Aaronson and Alex Arkhipov. The computational complexity of linear optics. In *Proceedings of the 43rd ACM Symposium on Theory of Computing*, pages 333–342, 2011. doi:10.1145/1993636.1993682.
2. Scott Aaronson and Alex Arkhipov. BosonSampling is far from uniform. *Quantum Information & Computation*, 14(15-16):1383–1423, 2014. URL: http://www.rintonpress.com/xxqic14/qic-14-1516/1383-1423.pdf.
3. Scott Aaronson and Lijie Chen. Complexity-Theoretic Foundations of Quantum Supremacy Experiments. In *Proceedings of the 32nd Computational Complexity Conference*, pages 22:1–22:67, 2017. doi:10.4230/LIPIcs.CCC.2017.22.
4. Andris Ambainis. Understanding quantum algorithms via query complexity. In *Proceedings of the 2018 International Congress of Mathematicians*, volume 3, pages 3249–3270, 2018.
5. Jonathan Barrett, Carlton M. Caves, Bryan Eastin, Matthew B. Elliott, and Stefano Pironio. Modeling Pauli measurements on graph states with nearest-neighbor classical communication. *Physical Review A*, 75:012103, 2007. doi:10.1103/PhysRevA.75.012103.
6. Adam Bene Watts, Robin Kothari, Luke Schaeffer, and Avishay Tal. Exponential separation between shallow quantum circuits and unbounded fan-in shallow classical circuits. In *Proceedings of the 43rd ACM Symposium on Theory of Computing*, 2019. To appear.
7. Ethan Bernstein and Umesh V. Vazirani. Quantum Complexity Theory. *SIAM Journal on Computing*, 26(5):1411–1473, 1997. doi:10.1137/S0097539796300921.

8 Adam Bouland, Bill Fefferman, Chinmay Nirkhe, and Umesh Vazirani. "Quantum Supremacy" and the Complexity of Random Circuit Sampling. In *Proceedings of the 10th Innovations in Theoretical Computer Science conference*, pages 15:1–15:2, 2019. arXiv:1803.04402. doi:10.4230/LIPIcs.ITCS.2019.15.

9 Sergey Bravyi, David Gosset, and Robert König. Quantum advantage with shallow circuits. arXiv:1704.00690 (preliminary version of [10]), 2017. arXiv:1704.00690.

10 Sergey Bravyi, David Gosset, and Robert König. Quantum advantage with shallow circuits. *Science*, 362(6412):308–311, 2018. doi:10.1126/science.aar3106.

11 Michael J. Bremner, Richard Jozsa, and Dan J. Shepherd. Classical simulation of commuting quantum computations implies collapse of the polynomial hierarchy. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 467(2126):459–472, 2010. doi:10.1098/rspa.2010.0301.

12 Michael J. Bremner, Ashley Montanaro, and Dan J. Shepherd. Average-Case Complexity Versus Approximate Simulation of Commuting Quantum Computations. *Physical Review Letters*, 117:080501, 2016. doi:10.1103/PhysRevLett.117.080501.

13 Michael J. Bremner, Ashley Montanaro, and Dan J. Shepherd. Achieving quantum supremacy with sparse and noisy commuting quantum circuits. *Quantum*, 1:8, 2017. doi:10.22331/q-2017-04-25-8.

14 Matthew Coudron, Jalex Stark, and Thomas Vidick. Trading locality for time: certifiable randomness from low-depth circuits, 2018. arXiv:1810.04233.

15 Edward Farhi and Aram W. Harrow. Quantum supremacy through the quantum approximate optimization algorithm, 2016. arXiv:1602.07674.

16 Keisuke Fujii, Hirotada Kobayashi, Tomoyuki Morimae, Harumichi Nishimura, Shuhei Tamate, and Seiichiro Tani. Impossibility of Classically Simulating One-Clean-Qubit Model with Multiplicative Error. *Physical Review Letters*, 120:200502, 2018. doi:10.1103/PhysRevLett.120.200502.

17 Keisuke Fujii and Shuhei Tamate. Computational quantum-classical boundary of noisy commuting quantum circuits. *Scientific Reports*, 6(25598), 2016. doi:10.1038/srep25598.

18 Frederic Green, Steven Homer, Cristopher Moore, and Christopher Pollett. Counting, fanout and the complexity of quantum ACC. *Quantum Information & Computation*, 2(1):35–65, 2002. URL: http://portal.acm.org/citation.cfm?id=2011420.

19 Marc Hein, Jens Eisert, and Hans J. Briegel. Multiparty entanglement in graph states. *Physical Review A*, 69:062311, 2004. doi:10.1103/PhysRevA.69.062311.

20 Peter Høyer and Robert Spalek. Quantum Fan-out is Powerful. *Theory of Computing*, 1(1):81–103, 2005. doi:10.4086/toc.2005.v001a005.

21 Attila Kondacs and John Watrous. On the Power of Quantum Finite State Automata. In *Proceedings of the 38th Annual Symposium on Foundations of Computer Science*, pages 66–75, 1997. doi:10.1109/SFCS.1997.646094.

22 François Le Gall, Harumichi Nishimura, and Ansis Rosmanis. Quantum Advantage for the LOCAL Model in Distributed Computing. In *Proceedings of the International Symposium on Theoretical Aspects of Computer Science*, pages 49:1–49:14, 2019. doi:10.4230/LIPIcs.STACS.2019.49.

23 Tomoyuki Morimae, Keisuke Fujii, and Joseph F. Fitzsimons. Hardness of Classically Simulating the One-Clean-Qubit Model. *Physical Review Letters*, 112:130502, 2014. doi:10.1103/PhysRevLett.112.130502.

24 Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000. doi:10.1017/CBO9780511976667.

25 Ran Raz and Avishay Tal. Oracle Separation of BQP and PH. *Proceedings of the 43rd ACM Symposium on Theory of Computing*, 2019. To appear.

26 Yasuhiro Takahashi and Seiichiro Tani. Collapse of the Hierarchy of Constant-Depth Exact Quantum Circuits. *Computational Complexity*, 25(4):849–881, 2016. doi:10.1007/s00037-016-0140-0.

27 Barbara M. Terhal and David P. DiVincenzo. Adaptive quantum computation, constant depth quantum circuits and Arthur-Merlin games. *Quantum Information & Computation*, 4(2):134–145, 2004. URL: http://portal.acm.org/citation.cfm?id=2011582.

Time-Space Lower Bounds for Two-Pass Learning

Sumegha Garg
Department of Computer Science, Princeton University, USA
sumegha.garg@gmail.com

Ran Raz
Department of Computer Science, Princeton University, USA
ran.raz.mail@gmail.com

Avishay Tal
Department of Computer Science, Stanford University, USA
avishay.tal@gmail.com

Abstract

A line of recent works showed that for a large class of learning problems, any learning algorithm requires either super-linear memory size or a super-polynomial number of samples [11, 7, 12, 9, 2, 5]. For example, any algorithm for learning parities of size n requires either a memory of size $\Omega(n^2)$ or an exponential number of samples [11].

All these works modeled the learner as a one-pass branching program, allowing only one pass over the stream of samples. In this work, we prove the first memory-samples lower bounds (with a super-linear lower bound on the memory size and super-polynomial lower bound on the number of samples) when the learner is allowed two passes over the stream of samples. For example, we prove that any two-pass algorithm for learning parities of size n requires either a memory of size $\Omega(n^{1.5})$ or at least $2^{\Omega(\sqrt{n})}$ samples.

More generally, a matrix $M : A \times X \to \{-1, 1\}$ corresponds to the following learning problem: An unknown element $x \in X$ is chosen uniformly at random. A learner tries to learn x from a stream of samples, $(a_1, b_1), (a_2, b_2) \ldots$, where for every i, $a_i \in A$ is chosen uniformly at random and $b_i = M(a_i, x)$.

Assume that k, ℓ, r are such that any submatrix of M of at least $2^{-k} \cdot |A|$ rows and at least $2^{-\ell} \cdot |X|$ columns, has a bias of at most 2^{-r}. We show that any two-pass learning algorithm for the learning problem corresponding to M requires either a memory of size at least $\Omega\left(k \cdot \min\{k, \sqrt{\ell}\}\right)$, or at least $2^{\Omega(\min\{k, \sqrt{\ell}, r\})}$ samples.

2012 ACM Subject Classification Theory of computation → Machine learning theory; Theory of computation → Circuit complexity

Keywords and phrases branching program, time-space tradeoffs, two-pass streaming, PAC learning, lower bounds

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.22

Related Version The paper is also available at `https://eccc.weizmann.ac.il/report/2019/071/`.

Funding *Ran Raz*: Research supported by the Simons Collaboration on Algorithms and Geometry, by a Simons Investigator Award and by the National Science Foundation grant No. CCF-1714779. *Avishay Tal*: Part of this work was done when the author was a member at the Institute for Advanced Study, Princeton, NJ. Research supported by the Simons Collaboration on Algorithms and Geometry and by the National Science Foundation grant No. CCF-1412958.

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka; Article No. 22; pp. 22:1–22:39

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

COMPUTATIONAL COMPLEXITY CONFERENCE

1 Introduction

A large number of recent works studied the problem of proving memory-samples lower bounds for learning [14, 15, 11, 16, 7, 8, 12, 9, 10, 2, 5, 4], a study that was initiated by the beautiful papers of Shamir [14] and Steinhardt, Valiant and Wager [15]. The motivation for studying this question comes from learning theory, computational complexity and cryptography (see for example the discussion and references in [14, 15, 11, 16, 7, 10]).

Steinhardt, Valiant and Wager conjectured that any algorithm for learning parities of size n requires either a memory of size $\Omega(n^2)$ or an exponential number of samples. This conjecture was proven in [11], followed by a line of works that showed that for a large number of learning problems, any learning algorithm requires either super-linear memory size or a super-polynomial number of samples [7, 12, 9, 2, 5]. For example, such bounds were established for learning sparse parities, linear-size DNF Formulas, linear-size Decision Trees and logarithmic-size Juntas [7]; learning low-degree polynomials [2, 5]; learning from sparse linear equations and low-degree polynomial equations [5]; learning codewords from random coordinates [12, 9, 5]; etc.

All previous memory-samples lower bounds (in the regime where the lower bound on the memory size is super-linear and the lower bound on the number of samples is super-polynomial) modeled the learning algorithm by a *one-pass branching program*, allowing only one pass over the stream of samples.

In this work, we prove the first such results when two passes over the stream of samples are allowed. (We remark that we leave open the question of handling more than two passes. While some parts of the current proof naturally extend to more than two passes, others are more delicate.)

Our Results

As in [12, 2, 7], we represent a learning problem by a matrix. Let X, A be two finite sets of size larger than 1 (where X represents the concept-class that we are trying to learn and A represents the set of possible samples). Let $M : A \times X \to \{-1, 1\}$ be a matrix. The matrix M represents the following learning problem: An unknown element $x \in X$ was chosen uniformly at random. A learner tries to learn x from a stream of samples, $(a_1, b_1), (a_2, b_2) \ldots$, where for every i, $a_i \in A$ is chosen uniformly at random and $b_i = M(a_i, x)$.

We model the learner for the learning problem that corresponds to the matrix M, by a *two-pass ordered branching program* (Definition 2). Such a program reads the entire stream of samples twice, in the exact same order. Roughly speaking, the model allows a learner with infinite computational power, and bounds only the memory size of the learner and the number of samples used.

As in [5], our result is stated in terms of the properties of the matrix M as a two-source extractor. Two-source extractors, first studied by Santha and Vazirani [13] and Chor and Goldreich [3], are central objects in the study of randomness and derandomization. As in [5], our results hold whenever the matrix M has (even relatively weak) two-source extractor properties.

Roughly speaking, our main result can be stated as follows: Assume that k, ℓ, r are such that any submatrix of M of at least $2^{-k} \cdot |A|$ rows and at least $2^{-\ell} \cdot |X|$ columns, has a bias of at most 2^{-r}. Then, any two-pass learning algorithm for the learning problem corresponding to M requires either a memory of size at least $\Omega\left(k \cdot \min\{k, \sqrt{\ell}\}\right)$, or at least $2^{\Omega(\min\{k, \sqrt{\ell}, r\})}$ samples.

Formally, our result is stated in Theorem 4 in terms of the properties of M as an L_2-Extractor (Definition 1), a notion that was defined in [5] and (as formally proved in [5]) is closely related to the notion of two-source extractor. (The two notions are equivalent up to small changes in the parameters.)

As in [5], our main result can be used to prove (two-pass) memory-samples lower bounds for many of the problems that were previously studied in this context. For example, for learning parities, sparse parities, DNFs, decision trees, random matrices, error correcting codes, etc. For example, our main result implies that any two-pass algorithm for learning parities of size n requires either a memory of size $\Omega(n^{1.5})$ or at least $2^{\Omega(\sqrt{n})}$ samples.

Related Work

To the best of our knowledge, the only previous work that proved memory-samples lower bounds for more than one pass over the stream of samples, is the intriguing recent work of Dagan and Shamir [4]. We note however that their results apply for a very different setting and regime of parameters, where the obtained lower bound on the number of samples is at most polynomial in the dimension of the problem. (Their result is proved in a very different setting, where the samples may be noisy, and the lower bound obtained on the number of samples is at most the product of the length of one sample times one over the information given by each sample).

Motivation and Discussion

Many previous works studied the resources needed for learning, under certain information, communication or memory constraints (see in particular [14, 15, 11, 16, 7, 8, 12, 9, 10, 2, 5, 4] and the many references given there). A main message of some of these works is that for some learning problems, access to a relatively large memory is crucial. In other words, in some cases, learning is infeasible, due to memory constraints.

From the point of view of human learning, such results may help to explain the importance of memory in cognitive processes. From the point of view of machine learning, these results imply that a large class of learning algorithms cannot learn certain concept classes. In addition, these works are related to computational complexity and have applications in bounded-storage cryptography.

Most of these works apply to bounded-memory learning algorithms that consider the samples one by one, with only one pass over the samples. In many practical situations, however, more than one pass over the samples is used, so it's desirable to extend these results to more than one pass over the samples.

From the point of view of computational complexity, the problem of extending these works to more than one pass over the samples is fascinating and challenging. It's a common practice in streaming-complexity to consider more than one pass over the inputs, and in computational complexity read-k-times branching programs have attracted a lot of attention.

We note that by Barrington's celebrated result, any function in NC can be computed by a polynomial-length branching program of width 5 [1]. Hence, proving super-polynomial lower bounds on the time needed for computing a function, by a branching program of width 5, with polynomially many passes over the input, would imply super-polynomial lower bounds for formula size, and is hence a very challenging problem.

Finally, let us mention that technically, allowing more than one pass over the samples is very challenging, as all previous techniques are heavily based on the fact that in the one-pass case all the samples are independent and hence at each time step, the learning algorithm has no information about the next sample that it is going to see.

Techniques

Our proof builds on the works of [12, 5] that gave a general technique for proving memory-samples lower bounds for learning problems. However, these works (as well as all other previous works that prove memory-samples lower bounds in this regime of parameters) are heavily based on the fact that in the one-pass case all the samples are independent and hence at each time step, the learning algorithm has no information about the next sample that it is going to see. Roughly speaking, the proofs of [12, 5] bound the L_2-norm of the distribution of x, conditioned on reaching a given vertex v of the branching program, but they rely on the fact that the next sample is independent of x. Once one allows more than one pass over the stream of samples, the assumption that the next sample is independent of x doesn't hold, as in the second pass the vertex may remember a lot of information about the joint distribution of x and $a_1, \ldots, a_m$.

Roughly speaking, [12, 5] considered the computation-path of the branching program and defined "stopping-rules". Intuitively, the computation stops if certain "bad" events occur. The proofs show that each stopping rule is only applied with negligible probability and that conditioned on the event that the computation didn't stop, the L_2-norm of the distribution of x, conditioned on reaching a vertex v of the branching program, is small (which implies that the program didn't learn x).

When more than one pass over the samples is allowed, there is a serious problem with this approach. After one pass, a vertex of the branching program has joint information on x and $a_1, \ldots, a_m$. If we only keep track of the distribution of x conditioned on that vertex, it could be the case that the next sample completely reveals x. One conceptual problem seems to be that the second part of the program (that is, the part that is doing the second pass) is not aware of what the first part did. An idea that turned out to be very important in our proof is to take the second part to be the product of the first and second part, so that, in some sense, the second part of the computation runs its own copy of the first part. In addition, we have each vertex in the second part remembering the vertex reached at the end of the first part.

As in [12, 5], we define stopping rules for the computation-path and we prove that the probability that the computation stops is small. We then analyze each part separately, as a read once program. For each part separately, we prove that conditioned on the event that the program didn't stop, the L_2-norm of the distribution of x, conditioned on reaching a vertex v, is small. It turns out that since the second part of the program runs its own copy of the first part, the analysis of each part separately is sufficient.

We note, however, that the entire proof is completely different than [12, 5]. The stopping rules are different and are defined differently for each part. The proof that the computation stops with low probability is much more delicate and complicated. The main challenge is that when analyzing the probability to stop on the second part, we cannot ignore the first part and we need to prove that we stop with low probability on the second part, when starting from the start vertex of the first part (that is, the start vertex of the entire program). This turns out to be very challenging and, in particular, requires a use of the results for one-pass branching programs.

A proof outline is given in Section 4.

2 Preliminaries

Denote by log the logarithm to base 2. For a random variable Z and an event E, we denote by $\mathbb{P}_Z$ the distribution of the random variables Z, and we denote by $\mathbb{P}_{Z|E}$ the distribution of the random variable Z conditioned on the event E.

We will sometimes take probabilities and expectations, conditioned on events E that may be empty. We think of these probabilities and expectations as 0, when the event E is empty.

2.1 Learning Problem

We represent a learning problem by a matrix. Let X, A be two finite sets of size larger than 1 (where X represents the concept-class that we are trying to learn and A represents the set of possible samples). Let $M : A \times X \to \{-1, 1\}$ be a matrix. The matrix M represents the following learning problem: An unknown element $x \in X$ was chosen uniformly at random. A learner tries to learn x from a stream of samples, $(a_1, b_1), (a_2, b_2) \ldots$, where for every i, $a_i \in A$ is chosen uniformly at random and $b_i = M(a_i, x)$.

Let $n = \log |X|$ and $n' = \log |A|$.

2.2 Norms and Inner Products

Let $p \geq 1$. For a function $f : X \to \mathbb{R}$, denote by $\|f\|_p$ the L_p norm of f, with respect to the uniform distribution over X, that is:

$$\|f\|_p = \left(\mathop{\mathbf{E}}_{x \in_R X} [|f(x)|^p] \right)^{1/p}.$$

For two functions $f, g : X \to \mathbb{R}$, define their inner product with respect to the uniform distribution over X as

$$\langle f, g \rangle = \mathop{\mathbf{E}}_{x \in_R X} [f(x) \cdot g(x)].$$

For a matrix $M : A \times X \to \mathbb{R}$ and a row $a \in A$, we denote by $M_a : X \to \mathbb{R}$ the function corresponding to the a-th row of M. Note that for a function $f : X \to \mathbb{R}$, we have $\langle M_a, f \rangle = \frac{(M \cdot f)_a}{|X|}$.

2.3 L_2-Extractors

▶ **Definition 1.** *L_2-Extractor: Let X, A be two finite sets. A matrix $M : A \times X \to \{-1, 1\}$ is a (k, ℓ)-L_2-Extractor with error 2^{-r}, if for every non-negative $f : X \to \mathbb{R}$ with $\frac{\|f\|_2}{\|f\|_1} \leq 2^{\ell}$ there are at most $2^{-k} \cdot |A|$ rows a in A with*

$$\frac{|\langle M_a, f \rangle|}{\|f\|_1} \geq 2^{-r}.$$

2.4 Computational Model

In the following definition, we model the learner for the learning problem that corresponds to the matrix M, by a *branching program*. We consider a q-pass ordered branching program. Such a program reads the entire input q times, in the exact same order. That is, the program has q parts (that are sequential in time). Each part reads the same stream in the exact same order. Our main result is proved for two-pass ordered branching programs, that is, for the case $q = 2$.

▶ **Definition 2.**

***q*-Pass Branching Program for a Learning Problem:** *A q-pass (ordered) branching program of length $q \cdot m$ and width d, for learning, is a directed (multi) graph with vertices arranged in $qm + 1$ layers containing at most d vertices each. In the first layer, that we think of as layer 0, there is only one vertex, called the start vertex. A vertex of outdegree 0 is called a leaf. All vertices in the last layer are leaves (but there may be additional leaves). Every non-leaf vertex in the program has $2|A|$ outgoing edges, labeled by elements $(a, b) \in A \times \{-1, 1\}$, with exactly one edge labeled by each such (a, b), and all these edges going into vertices in the next layer. Each leaf v in the program is labeled by an element $\tilde{x}(v) \in X$, that we think of as the output of the program on that leaf.*

Computation-Path: *The samples $(a_1, b_1), \ldots, (a_m, b_m) \in A \times \{-1, 1\}$ that are given as input define a computation-path in the branching program, by starting from the start vertex and following at step $(j-1) \cdot m + i$ the edge labeled by (a_i, b_i) (where $j \in [q]$ and $i \in [m]$), until reaching a leaf. The program outputs the label $\tilde{x}(v)$ of the leaf v reached by the computation-path.*

Success Probability: *The success probability of the program is the probability that $\tilde{x} = x$, where $\tilde{x}$ is the element that the program outputs, and the probability is over $x, a_1, \ldots, a_m$ (where x is uniformly distributed over X and $a_1, \ldots, a_m$ are uniformly distributed over A, and for every i, $b_i = M(a_i, x)$).*

Remark: *We will sometimes consider branching programs in which the leaves are not labeled, and hence the program doesn't return any value. It will be convenient to refer to such objects also as branching programs. In particular, we will view a part of the branching program (e.g., the first few layers of a program) also as a branching program.*

We think of the program as composed of q parts, where for every $j \in [q]$, part-j contains layers $\{(j-1) \cdot m + i\}_{i \in [m]}$.

For convenience, we think of each vertex u of the branching program as having a small memory S_u that contains some information about the path that led to the vertex, that the vertex "remembers" (or "records"). Formally, this means that in the actual branching program the vertex u is split into distinct vertices $u_1, \ldots, u_{d(u)}$, according to the content of the memory S_u. Adding information to S_u means that the vertex u is further split into distinct vertices, according to the content of the information that was added. Thus, when we refer to a vertex u of a program, we mean, a vertex u plus content of the memory S_u.

In this paper, we will have the property that whenever we add some information to the memory of a vertex u, that information is never removed/forgotten. That is, information that was added to the memory of u, remains in the memory of all the vertices that can be reached from u.

As mentioned above, in this paper we focus on the case $q = 2$. We denote by v_0 the start vertex of the program and by v_1 the vertex reached at the end of the first part, that is, layer-m. Note that v_1 is a random variable that depends on $x, a_1, \ldots, a_m$.

2.5 Product of Programs

Intuitively, the product of two branching programs is a branching program that runs both programs in parallel.

▶ **Definition 3.**

Product of One-Pass Branching Programs: *Let B, B' be two one-pass branching programs for learning, of length m and widths d, d', respectively. The product $B \times B'$ is a (one-pass) branching program of length m and width $d \cdot d'$, as follows: For every $i \in \{0, \ldots, m\}$ and*

vertices v in layer-i of B and v' in layer-i of B', we have a vertex (v,v') in layer-i of $B \times B'$. For every two edges: (u,v) from layer-$(i-1)$ to layer-i of B and (u',v') from layer-$(i-1)$ to layer-i of B', both labeled by the same (a,b), we have in $B \times B'$ an edge $((u,u'),(v,v'))$ labeled by (a,b).

The label of a leaf (v,v') is the label given by the second program B'. The content of the memory $S_{(v,v')}$ of a vertex (v,v') is the concatenation of the content of S_v and the content of $S_{v'}$.

Remark: *We will use this definition also in cases where the leaves of B and/or B' are not labeled (that is, where B and/or B' do not output any value; see a remark in Definition 2).*

3 Main Result

Fix $k,\ell,r \in \mathbb{N}$, such that $\frac{r}{k}, \frac{r}{\ell}$ are smaller than a sufficiently small constant and $k < n'$, $\ell < n$. Let $\epsilon > 0$ be a sufficiently small constant. In particular, we assume that ϵ is sufficiently smaller than all other constants that we discuss, say, $\epsilon < \frac{1}{10^{10}}$. We assume that n, n' are sufficiently large. Let

$$\tilde{\ell} = \min\left\{k, \sqrt{\ell}\right\}.$$

Let

$$\tilde{r} = \min\left\{\tfrac{r}{100}, \tfrac{\tilde{\ell}}{100}\right\}. \tag{1}$$

We assume that

$$\tilde{r} > 100 \cdot \max\left\{\log n, \log n'\right\}. \tag{2}$$

We assume that M is a $(10k, 10\ell)$-L_2-extractor with error 2^{-10r}.

▶ **Theorem 4.** *Let X, A be two finite sets. Let $n = \log_2 |X|$ and $n' = \log_2 |A|$. Fix $k,\ell,r \in \mathbb{N}$, such that, $\frac{r}{k}, \frac{r}{\ell} < \frac{1}{100}$, and $k < n'$, $\ell < n$. Let $\epsilon > 0$ be a sufficiently small constant, say, $\epsilon < \frac{1}{10^{10}}$. Assume that n, n' are sufficiently large. Let*

$$\tilde{\ell} = \min\left\{k, \sqrt{\ell}\right\}.$$

Let

$$\tilde{r} = \min\left\{\tfrac{r}{100}, \tfrac{\tilde{\ell}}{100}\right\}.$$

Assume that

$$\tilde{r} > 100 \cdot \max\left\{\log n, \log n'\right\}.$$

Let $M : A \times X \to \{-1,1\}$ be a matrix which is a $(10k, 10\ell)$-L_2-extractor with error 2^{-10r}. Let B be a two-pass ordered branching program of length $2 \cdot m$, where m is at most $2^{\epsilon\tilde{r}}$, and width at most $d = 2^{\epsilon k\tilde{\ell}/10}$, for the learning problem that corresponds to the matrix M. Then, the success probability of B is at most $\frac{1}{100} + o(1)$.

4 Overview of the Proof

One-Pass Learners

We will start with giving a short outline of the proof of [12, 5] for one-pass learners. Assume that M is a $(10k, 10\ell)$-L_2-extractor with error 2^{-10r}, where $r < k, \ell$. Let B be a one-pass branching program for the learning problem that corresponds to the matrix M. Assume for a contradiction that B is of length $m = 2^{\epsilon r}$ and width $d = 2^{\epsilon k\ell}$, where ϵ is a small constant.

We define the *truncated-path*, $\mathcal{T}$, to be the same as the computation-path of B, except that it sometimes stops before reaching a leaf. Roughly speaking, $\mathcal{T}$ stops before reaching a leaf if certain "bad" events occur. Nevertheless, we show that the probability that $\mathcal{T}$ stops before reaching a leaf is negligible, so we can think of $\mathcal{T}$ as almost identical to the computation-path.

For a vertex v of B, we denote by E_v the event that $\mathcal{T}$ reaches the vertex v. We denote by $\Pr(v) = \Pr(E_v)$ the probability for E_v (where the probability is over $x, a_1, \ldots, a_m$), and we denote by $\mathbb{P}_{x|v} = \mathbb{P}_{x|E_v}$ the distribution of the random variable x conditioned on the event E_v. Similarly, for an edge e of the branching program B, let E_e be the event that $\mathcal{T}$ traverses the edge e. Denote, $\Pr(e) = \Pr(E_e)$, and $\mathbb{P}_{x|e} = \mathbb{P}_{x|E_e}$.

A vertex v of B is called *significant* if

$$\left\|\mathbb{P}_{x|v}\right\|_2 > 2^{\ell} \cdot 2^{-n}.$$

Roughly speaking, this means that conditioning on the event that $\mathcal{T}$ reaches the vertex v, a non-negligible amount of information is known about x. In order to guess x with a non-negligible success probability, $\mathcal{T}$ must reach a significant vertex. We show that the probability that $\mathcal{T}$ reaches any significant vertex is negligible, and thus the main result follows.

To prove this, we show that for every fixed significant vertex s, the probability that $\mathcal{T}$ reaches s is at most $2^{-\Omega(k\ell)}$ (which is smaller than one over the number of vertices in B). Hence, we can use a union bound to prove the bound.

The proof that the probability that $\mathcal{T}$ reaches s is extremely small is the main part of the proof. To that end, we use the following functions to measure the progress made by the branching program towards reaching s.

Let L_i be the set of vertices v in layer-i of B, such that $\Pr(v) > 0$. Let Γ_i be the set of edges e from layer-$(i-1)$ of B to layer-i of B, such that $\Pr(e) > 0$. Let

$$\mathcal{Z}_i = \sum_{v \in L_i} \Pr(v) \cdot \langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s} \rangle^k,$$

$$\mathcal{Z}'_i = \sum_{e \in \Gamma_i} \Pr(e) \cdot \langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle^k.$$

We think of $\mathcal{Z}_i, \mathcal{Z}'_i$ as measuring the progress made by the branching program, towards reaching a state with distribution similar to $\mathbb{P}_{x|s}$.

We show that each $\mathcal{Z}_i$ may only be negligibly larger than $\mathcal{Z}_{i-1}$. Hence, since it's easy to calculate that $\mathcal{Z}_0 = 2^{-2nk}$, it follows that $\mathcal{Z}_i$ is close to 2^{-2nk}, for every i. On the other hand, if s is in layer-i then $\mathcal{Z}_i$ is at least $\Pr(s) \cdot \langle \mathbb{P}_{x|s}, \mathbb{P}_{x|s} \rangle^k$. Thus, $\Pr(s) \cdot \langle \mathbb{P}_{x|s}, \mathbb{P}_{x|s} \rangle^k$ cannot be much larger than 2^{-2nk}. Since s is significant, $\langle \mathbb{P}_{x|s}, \mathbb{P}_{x|s} \rangle^k > 2^{\ell k} \cdot 2^{-2nk}$ and hence $\Pr(s)$ is at most $2^{-\Omega(k\ell)}$.

The proof that $\mathcal{Z}_i$ may only be negligibly larger than $\mathcal{Z}_{i-1}$ is done in two steps. We show by a simple convexity argument that $\mathcal{Z}_i \leq \mathcal{Z}'_i$. The hard part is to prove that $\mathcal{Z}'_i$ may only be negligibly larger than $\mathcal{Z}_{i-1}$.

For this proof, we define for every vertex v, the set of edges $\Gamma_{out}(v)$ that are going out of v, such that $\Pr(e) > 0$ and show that for every vertex v,

$$\sum_{e \in \Gamma_{out}(v)} \Pr(e) \cdot \langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle^k$$

may only be negligibly higher than

$$\Pr(v) \cdot \langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s} \rangle^k.$$

For this proof, we consider the function $\mathbb{P}_{x|v} \cdot \mathbb{P}_{x|s}$. We first show how to bound $\left\|\mathbb{P}_{x|v} \cdot \mathbb{P}_{x|s}\right\|_2$. We then consider two cases: If $\left\|\mathbb{P}_{x|v} \cdot \mathbb{P}_{x|s}\right\|_1$ is negligible, then $\langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s} \rangle^k$ is negligible and doesn't contribute much, and we show that for every $e \in \Gamma_{out}(v)$, $\langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle^k$ is also negligible and doesn't contribute much. If $\left\|\mathbb{P}_{x|v} \cdot \mathbb{P}_{x|s}\right\|_1$ is non-negligible, we use the bound on $\left\|\mathbb{P}_{x|v} \cdot \mathbb{P}_{x|s}\right\|_2$ and the assumption that M is a $(10k, 10\ell)$-L_2-extractor to show that for almost all edges $e \in \Gamma_{out}(v)$, we have that $\langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle^k$ is very close to $\langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s} \rangle^k$. Only an exponentially small (2^{-k}) fraction of edges are "bad" and give a significantly larger $\langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle^k$.

The reason that in the definitions of $\mathcal{Z}_i$ and $\mathcal{Z}'_i$ we raised $\langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s} \rangle$ and $\langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle$ to the power of k is that this is the largest power for which the contribution of the "bad" edges is still small (as their fraction is 2^{-k}).

This outline oversimplifies many details. Let us briefly mention two of them. First, it is not so easy to bound $\left\|\mathbb{P}_{x|v} \cdot \mathbb{P}_{x|s}\right\|_2$. We do that by bounding $\left\|\mathbb{P}_{x|s}\right\|_2$ and $\left\|\mathbb{P}_{x|v}\right\|_\infty$. In order to bound $\left\|\mathbb{P}_{x|s}\right\|_2$, we force $\mathcal{T}$ to stop whenever it reaches a significant vertex (and thus we are able to bound $\left\|\mathbb{P}_{x|v}\right\|_2$ for every vertex reached by $\mathcal{T}$). In order to bound $\left\|\mathbb{P}_{x|v}\right\|_\infty$, we force $\mathcal{T}$ to stop whenever $\mathbb{P}_{x|v}(x)$ is large, which allows us to consider only the "bounded" part of $\mathbb{P}_{x|v}$. (This is related to the technique of *flattening* a distribution that was used in [6]). Second, some edges are so "bad" that their contribution to $\mathcal{Z}'_i$ is huge so they cannot be ignored. We force $\mathcal{T}$ to stop before traversing any such edge. (This is related to an idea that was used in [7] of analyzing separately paths that traverse "bad" edges). We show that the total probability that $\mathcal{T}$ stops before reaching a leaf is negligible.

Thus, in [12, 5] there are three stopping rules: We stop if we reach a **significant vertex**. We stop if we have a **bad edge** and we stop if x is a **significant-value** of $\mathbb{P}_{x|v}$, that is, if $\mathbb{P}_{x|v}(x)$ is too large.

Two-Pass Learners

Let us now give a short outline of the additional ideas in the proof for two-pass learners. Let B be a two-pass branching program for the learning problem that corresponds to the matrix M. We denote by v_0 the starting vertex of the program and by v_1 the vertex reached at the end of the first part. We assume without loss of generality that the answers are given in the last layer of the program.

We update the second part so that every vertex v in the second part "remembers" v_1. This information is stored in the memory S_v. Formally, this means that starting from every possible v_1, we have a separate copy of the entire second part of the program. We then change the second part so that it is now the **product** (see definition 2) of the first part and the second part. Intuitively, this means that the second part runs a copy of the first part of the computation, in parallel to its own computation.

As in [12, 5], we define the *truncated-path*, $\mathcal{T}$, to be the same as the computation-path of the new branching program, except that it sometimes stops before reaching a leaf. Roughly speaking, $\mathcal{T}$ stops before reaching a leaf if certain "bad" events occur. Nevertheless, we show

that the probability that $\mathcal{T}$ stops before reaching a leaf is small, so we can think of $\mathcal{T}$ as essentially identical to the computation-path. The decision of whether or not $\mathcal{T}$ stops on a given vertex v in layer-i of part-j will depend on v, S_v, x, a_{i+1}. For that reason, we are able to consider the path $\mathcal{T}$, starting from any vertex v (without knowing the history of the path that led to v, except for the information stored in S_v).

Let v be a vertex in the second part of the program (where an answer should be given). The vertex v remembers (in S_v) the vertex v_1. We denote by $v_1 \to v$ the event that the path $\mathcal{T}$ that starts from v_1 reaches v (where v_1 is the vertex at the end of the first part of the program that v remembers, and the event is over $x, a_1, \ldots, a_m$). We denote by $v_0 \to v$ the event that the path $\mathcal{T}$ that starts from the start vertex v_0 reaches v. More generally, for two vertices w_1, w_2 in the program, we denote by $w_1 \to w_2$ the event (over $x, a_1, \ldots, a_m$) that the path $\mathcal{T}$ that starts from w_1 reaches w_2.

Let v be a vertex in the last layer of the program, such that $\Pr(v_0 \to v) > 0$. Since v remembers v_1, the event $v_0 \to v$ is equivalent to $v_0 \to v_1 \to v$ (where v_1 is the vertex remembered by v). Since the second part of the program runs a copy of the first part and since v is in the last layer, the event $v_1 \to v$ implies the event $v_0 \to v_1$. Thus, the event $v_0 \to v$ is equivalent to $v_1 \to v$.

Moreover, this is true when conditioning on x, and hence,

$$\mathbb{P}_{x|v_0 \to v} = \mathbb{P}_{x|v_1 \to v}$$

and

$$\Pr[v_0 \to v] = \Pr[v_1 \to v].$$

This is a crucial point as it means that

$$\left\|\mathbb{P}_{x|v_0 \to v}\right\|_2 = \left\|\mathbb{P}_{x|v_1 \to v}\right\|_2,$$

that is, if we bound $\left\|\mathbb{P}_{x|v_1 \to v}\right\|_2$ we also get a bound on $\left\|\mathbb{P}_{x|v_0 \to v}\right\|_2$.

The bound on $\left\|\mathbb{P}_{x|v_0 \to v}\right\|_2$ is what we really need because if this is small then the program cannot answer correctly. On the other hand, the bound on $\left\|\mathbb{P}_{x|v_1 \to v}\right\|_2$ is easier to obtain as it is a bound for a one-pass branching program. Thus, all we need is a bound on $\left\|\mathbb{P}_{x|v_1 \to v}\right\|_2$, which is a bound for a one-pass branching program, and we already know how to obtain bounds on the conditional distribution for one-pass programs.

Things, however, are not so simple, as we need to prove that $\mathcal{T}$ stops with small probability, when starting from v_0, rather than v_1. The main problem with using the previous stopping rules (in the second part of the program) is that it's impossible to prove that we stop on a bad edge with negligible probability (as demonstrated next). Roughly speaking, we say that an edge $e = (u, v)$ is "bad" if the equation on it splits the distribution $\mathbb{P}_{x|u}$ in a biased way. That is, a good edge is one where roughly half the probability mass of $\mathbb{P}_{x|u}$ satisfies the equation on the edge e. If the program stores in memory the i-th sample from the first pass, then in the i-th step of the second pass, an edge $e = (u, v)$ will definitely be bad, since it will not split the distribution $\mathbb{P}_{x|u}$ evenly.

For that reason, we change the bad-edges stopping rule. We say that an edge (v, u), labelled by (a, b), is of *high probability* if the probability to sample a, conditioning on reaching v from v_0 (that is, reaching v from the starting vertex of the entire program) is large. The third stopping rule is changed so that $\mathcal{T}$ doesn't stop on a bad edge if it is of high probability. Instead, if $\mathcal{T}$ traverses such an edge, we "remember" the time step in which $\mathcal{T}$ traversed that edge, in all the future. That is, we enter the index i to S_u (and remember it in all the future,

until the end of the program). In addition, we add a stopping rule that stops if the edge is "very-bad" and a stopping rule that stops if the number of indices in S_v is too large, that is, if the number of high-probability edges that were already traversed is too large (intuitively, S_v won't be too large because of the bounded memory size).

We analyze separately the probability to stop because of each stopping rule. The main challenge is that we need to analyze these probabilities when starting from v_0, that is, when running a two-pass program. These proofs are technically hard, but the main reason that we manage to analyze these probabilities is the following:

Recall that the second part of the program runs a copy of the first part of the program. Thus, a vertex v in layer-i of the second part has a corresponding vertex v' in layer-i of the first part, such that, if the path $\mathcal{T}$ reached v it previously reached v'. Recall also that v remembers v_1, so if the path $\mathcal{T}$ reached v it previously reached v_1. Thus, the event $v_0 \to v$ is equivalent to $v_0 \to v' \to v_1 \to v$, that is, the event

$$(v_0 \to v') \wedge (v' \to v_1) \wedge (v_1 \to v).$$

Since the second part of the program runs a copy of the first part, the event $v_1 \to v$ implies the event $v_0 \to v'$. Hence, the event $v_0 \to v$ is equivalent to the event

$$(v' \to v_1) \wedge (v_1 \to v).$$

Note that v' is in layer-i of the first part and v is in layer-i of the second part, and from layer-i of the first part to layer-i of the second part, the program is a one-pass program and is hence easier to analyze.

5 Proof of Theorem 4

Assume that we have a two-pass ordered branching program, B, for the learning problem that corresponds to the matrix M. We assume without loss of generality that the output is given in the last layer. Assume that the length of the program is $2 \cdot m$, where m is at most $2^{\epsilon \tilde{r}}$ and the width of the program is at most $d = 2^{\epsilon k \tilde{\ell}/10}$. We will show that the success probability of B is at most $\frac{1}{100} + o(1)$.

Let

$$\ell_1 = \frac{\tilde{\ell}}{100}$$

and

$$\ell_2 = \ell.$$

5.1 The Truncated Path

Below, we will make some changes in the branching program B. We will denote by $\hat{B}$ the resulting branching program. Let v_0 be the start vertex of $\hat{B}$. We will denote by v_1 the vertex reached at the end of the first part of $\hat{B}$. Note that v_1 is a random variable, that depends on $x, a_1, \ldots, a_m$.

In the resulting branching program $\hat{B}$, we will have the property that the vertex v_1 reached at the end of the first part of the program is remembered by every future vertex v. That is, every vertex v, in the second part of the program, remembers which vertex the path that led to v reached, at the end of the first part of the program. Formally, this information is stored in S_v.

Below, we will define the *truncated-path*, $\mathcal{T}$, to be the same as the computation-path of the new branching program, except that it sometimes stops before reaching a leaf. Roughly speaking, $\mathcal{T}$ stops before reaching a leaf if certain "bad" events occur. Nevertheless, we show that the probability that $\mathcal{T}$ stops before reaching a leaf is small, so we can think of $\mathcal{T}$ as essentially identical to the computation-path. The decision of whether or not $\mathcal{T}$ stops on a given vertex v in layer-i of part-j will depend on v, S_v, x, a_{i+1}. For that reason, we are able to consider the path $\mathcal{T}$, starting from any vertex v (without knowing the history of the path that led to v, except for the information stored in S_v).

Let v be a vertex in the second part of the program. The vertex v remembers (in S_v) the vertex v_1. We denote by $v_1 \to v$ the event that the path $\mathcal{T}$ that starts from v_1 reaches v (where v_1 is the vertex at the end of the first part of the program that v remembers, and the event is over $x, a_1, \ldots, a_m$).

More generally, for two vertices w_1, w_2 in the program, we denote by $w_1 \to w_2$ the event (over $x, a_1, \ldots, a_m$) that the path $\mathcal{T}$ that starts from w_1 reaches w_2. In particular, $v_0 \to v$ is the event that the path $\mathcal{T}$ that starts from the start vertex v_0 reaches v.

We change the original branching program B as follows:

First Part

We define stopping rules for the first part as defined in [12, 5] for one-pass programs, as if the first part were the entire program. Next, we describe these rules formally.

Significant Vertices

We say that a vertex v in layer-i of the first part of the program is **significant** if

$$\left\|\mathbb{P}_{x|v_0\to v}\right\|_2 > 2^{\ell_1} \cdot 2^{-n}.$$

Significant Values

Even if v is not significant, $\mathbb{P}_{x|v_0\to v}$ may have relatively large values. For a vertex v in layer-i of the first part of the program, denote by $\mathrm{Sig}(v)$ the set of all $x' \in X$, such that,

$$\mathbb{P}_{x|v_0\to v}(x') > 2^{4\ell} \cdot 2^{-n}.$$

Bad Edges

For a vertex v in layer-i of the first part of the program, denote by $\mathrm{Bad}(v)$ the set of all $a \in A$, such that,

$$\left|(M \cdot \mathbb{P}_{x|v_0\to v})(a)\right| \geq 2^{-2r}.$$

The Truncated-Path $\mathcal{T}$ on the First Part

We define $\mathcal{T}$ on the first part, by induction on the layers. Assume that we already defined $\mathcal{T}$ until it reaches a vertex v in layer-i of the first part. The path $\mathcal{T}$ stops on v if (at least) one of the following occurs:

1. v is significant.
2. $x \in \mathrm{Sig}(v)$.
3. $a_{i+1} \in \mathrm{Bad}(v)$.

Otherwise, (unless $i = m$) $\mathcal{T}$ proceeds by following the edge labeled by (a_{i+1}, b_{i+1}) (same as the computational-path).

Second Part

We denote by v_1 the vertex in layer-m (that is, the last layer of the first part of the program) that is reached by $\mathcal{T}$. Note that v_1 is a random variable that depends on $x, a_1, \ldots, a_m$. We denote by d_1 the number of vertices in layer-m. We assume without loss of generality that each vertex in layer-m is reached with probability of at least $2^{-10\tilde{r}} \cdot d_1^{-1}$, as vertices reached with negligible probability can be ignored. Formally, if we reach a vertex in layer-m, such that, the probability to reach that vertex is smaller than $2^{-10\tilde{r}} \cdot d_1^{-1}$, the path $\mathcal{T}$ stops.

We update the second part so that every vertex v in the second part "remembers" v_1. This information is stored in the memory S_v. Formally, this means that starting from every possible v_1, we have a separate copy of the entire second part of the program.

We then change the second part so that it is now the **product** (see definition 2) of the first part (after it was changed as described above) and the second part. Intuitively, this means that the second part runs a copy of the first part of the computation, in parallel to its own computation.

Next, we define stopping rules for the second part, by induction over the layers, and at the same time (by the same induction), we also define for each vertex v, a list L_v of indices $i_1, \ldots, i_{d(v)} \in [m]$ that the vertex v remembers (that is, the list L_v is stored in the memory S_v). Once an index was added to L_v, it is remembered in all the future, that is, for every vertex u reached from v (in the second part of the program), we have $L_v \subseteq L_u$. Note that the stopping rules are defined for the updated second part (as described above).

The stopping rules for the second part extend the stopping rules in the case of one-pass programs, as defined in [12, 5], as if the second part were the entire program, with starting vertex v_1. However, the third stopping rule (*bad* edges) is now different. We say that an edge (v, u), labelled by (a, b), is of *high probability* if the probability to sample a, conditioning on reaching v from v_0 (that is, reaching v from the starting vertex of the entire program) is larger than $2^k \cdot 2^{-n'}$. That is, if v is in layer-i of the second part, (v, u) is of high probability if $\Pr[a_{i+1} = a \mid v_0 \to v] \geq 2^k \cdot 2^{-n'}$. The third stopping rule is changed so that $\mathcal{T}$ doesn't stop on a bad edge if it is of high probability. Instead, if $\mathcal{T}$ traverses such an edge, we "remember" the time step in which $\mathcal{T}$ traversed that edge, in all the future. That is, we enter the index i to L_u (and remember it in all the future, until the end of the program). In addition, we add a stopping rule that stops if the edge is "very-bad" and a stopping rule that stops if the number of indices in L_v is too large, that is, if the number of high-probability edges that were already traversed is too large.

Next, we describe these rules formally. We initiate $L_{v_1} = \emptyset$.

Significant Vertices

We say that a vertex v in layer-i of the second part of the program is **significant** if

$$\left\| \mathbb{P}_{x|v_1 \to v} \right\|_2 > 2^{\ell_2} \cdot 2^{-n}.$$

Significant Values

For a vertex v in layer-i of the second part of the program, denote by $\mathrm{Sig}(v)$ the set of all $x' \in X$, such that,

$$\mathbb{P}_{x|v_1 \to v}(x') > 2^{4\ell} \cdot 2^{-n}.$$

Bad Edges

For a vertex v in layer-i of the second part of the program, denote by $\mathrm{Bad}(v)$ the set of all $a \in A$, such that,

$$\left|(M \cdot \mathbb{P}_{x|v_1 \to v})(a)\right| \geq 2^{-2r}.$$

Very-Bad Edges

For a vertex v in layer-i of the second part of the program, denote by $\mathrm{VeryBad}(v)$ the set of all $(a,b) \in A \times \{-1,1\}$, such that,

$$\Pr_x[M(a,x) = b \mid v_1 \to v] \leq 2^{-4\tilde{\ell}}.$$

High-Probability Edges

For a vertex v in layer-i of the second part of the program, denote by $\mathrm{High}(v)$ the set of all $a \in A$, such that,

$$\Pr[a_{i+1} = a \mid v_0 \to v] \geq 2^k \cdot 2^{-n'}.$$

The Truncated-Path $\mathcal{T}$ on the Second Part

We define $\mathcal{T}$ on the second part, by induction on the layers. Assume that we already defined $\mathcal{T}$ until it reaches a vertex v in layer-i of the second part (and we already defined L_v). The path $\mathcal{T}$ stops on v if (at least) one of the following occurs:

1. v is significant.
2. $x \in \mathrm{Sig}(v)$.
3. $a_{i+1} \in \mathrm{Bad}(v) \setminus \mathrm{High}(v)$.
4. $(a_{i+1}, b_{i+1}) \in \mathrm{VeryBad}(v)$.
5. $|L_v| \geq 200\epsilon\tilde{\ell}$.
6. Recall that we changed the second part of the program so that it is the product of the first part and the (original) second part. This means that the second part of the program runs its own copy of the first part of the program. If the path $\mathcal{T}$, that was defined for the first part, stops on the copy of the first part that the second part runs, the path $\mathcal{T}$ stops on the vertex v too.

   ▶ Remark 5. We note that if $\mathcal{T}$ stopped on the first part, it couldn't have reached v_1 in the first place. Thus, conditioned on the event $v_0 \to v_1$, the path $\mathcal{T}$ didn't stop on the first part. Therefore, conditioned on the event $v_0 \to v_1$, the path $\mathcal{T}$ never stops because of stopping rule 6. Thus, this stopping rule is not necessary. Nevertheless, we add this stopping rule for completeness, so that it would be possible to consider the path $\mathcal{T}$ starting from any vertex (even in the middle of the program), without conditioning on the event of reaching that vertex.

Otherwise, unless $\mathcal{T}$ already reached the end of the second part, $\mathcal{T}$ proceeds by following the edge labeled by (a_{i+1}, b_{i+1}) (same as the computational-path). Let (v,u) be the edge labeled by (a_{i+1}, b_{i+1}). It remains to define the list L_u.

Updating L_u

Let (v,u) be the traversed edge, labeled by (a_{i+1}, b_{i+1}). If the traversed edge (v,u) is not a high-probability edge, that is, if $a_{i+1} \notin \mathrm{High}(v)$, we define $L_u = L_v$.

If the traversed edge (v, u) is a high-probability edge, that is, if $a_{i+1} \in \mathrm{High}(v)$, we define $L_u = L_v \cup \{i+1\}$, and hence, by induction, L_u is the list of all indices corresponding to the high-probability edges that $\mathcal{T}$ traversed, in the second part of the program, until reaching u.

5.2 Bounding the Width of the Branching Program $\hat{B}$

From now on, we will only consider the final branching program, $\hat{B}$.

The final branching program, $\hat{B}$, has a larger width than the original one. The main contributions to the larger width is that we changed the second part to be the product of the first and second parts of the original program and that each vertex in the second part of $\hat{B}$ remembers the vertex v_1 (reached at the end of the first part). This multiplies the memory needed (that is, the logarithm of the width of the program) by a factor of at most 3. In addition, each vertex v has to remember L_v, but by Equation (1) and since $\mathcal{T}$ stops when $|L_v| \geq 200\epsilon\tilde{\ell}$, this adds memory of at most $\frac{\epsilon\tilde{\ell}r}{100}$. Thus, the final width of $\hat{B}$ is at most $2^{\epsilon k\tilde{\ell}/2}$.

5.3 The Probability that $\mathcal{T}$ Stops is Small

We will now prove that the probability that $\mathcal{T}$ stops before reaching a leaf is at most $\frac{1}{100} + o(1)$.

▶ **Lemma 6.** *The probability that $\mathcal{T}$ stops before reaching a leaf is at most $\frac{1}{100} + o(1)$.*

Proof. First, recall that if $\mathcal{T}$ reaches a vertex in layer-m, such that, the probability to reach that vertex is smaller than $2^{-10\tilde{r}} \cdot d_1^{-1}$, then $\mathcal{T}$ stops. By the union bound, the probability that $\mathcal{T}$ stops because of this rule is at most $2^{-10\tilde{r}} = o(1)$.

We will now bound the probability that $\mathcal{T}$ stops because of each of the other stopping rules.

Recall that for two vertices w_1, w_2 in the program, we denote by $w_1 \to w_2$ the event (over $x, a_1, \ldots, a_m$) that the path $\mathcal{T}$ that starts from w_1 reaches w_2.

5.3.1 Stopping Rule 1: Significant-Vertices

▶ **Lemma 7.** *The probability that $\mathcal{T}$ reaches a significant vertex is at most $o(1)$.*

Lemma 7 is proved in Section 6.

Next, we will bound the probability that $\mathcal{T}$ stops because of each of the other stopping rules. By Lemma 7, it's sufficient to bound these probabilities, under the assumption that $\mathcal{T}$ doesn't reach any significant vertex (as otherwise, $\mathcal{T}$ would have stopped because of stopping rule 1).

5.3.2 Stopping Rule 2: Significant-Values

We will now bound the probability that $\mathcal{T}$ stops because of stopping rule 2. We will first prove the following claim.

▷ Claim 8. If v is a non-significant vertex in layer-i of part-j (where $j \in \{1, 2\}$), then

$$\Pr_x[x \in \mathrm{Sig}(v) \mid v_{j-1} \to v] \leq 2^{-2\ell}.$$

Proof. Since v is not significant,

$$\mathop{\mathbf{E}}_{x' \sim \mathbb{P}_{x|v_{j-1}\to v}} \left[\mathbb{P}_{x|v_{j-1}\to v}(x')\right] = \sum_{x' \in X} \left[\mathbb{P}_{x|v_{j-1}\to v}(x')^2\right] = 2^n \cdot \mathop{\mathbf{E}}_{x' \in_R X} \left[\mathbb{P}_{x|v_{j-1}\to v}(x')^2\right] \leq 2^{2\ell_j} \cdot 2^{-n}.$$

Hence, by Markov's inequality,

$$\Pr_{x' \sim \mathbb{P}_{x|v_{j-1}\to v}}\left[\mathbb{P}_{x|v_{j-1}\to v}(x') > 2^{4\ell}\cdot 2^{-n}\right] \le 2^{2\ell_j - 4\ell} \le 2^{-2\ell}.$$

Since conditioned on the event $v_{j-1} \to v$, the distribution of x is $\mathbb{P}_{x|v_{j-1}\to v}$, we obtain

$$\Pr_x\left[x \in \mathrm{Sig}(v) \;\middle|\; v_{j-1}\to v\right] = \Pr_x\left[\left(\mathbb{P}_{x|v_{j-1}\to v}(x) > 2^{4\ell}\cdot 2^{-n}\right) \;\middle|\; v_{j-1}\to v\right] \le 2^{-2\ell}. \qquad \blacktriangleleft$$

By Claim 8, if v is a non-significant vertex in layer-i of part-j then

$$\Pr_x[x \in \mathrm{Sig}(v) \mid v_{j-1}\to v] \le 2^{-2\ell} \le 2^{-4\tilde{\ell}}. \tag{3}$$

We need to bound from above

$$\mathop{\mathbf{E}}_v\left[\Pr_x[x \in \mathrm{Sig}(v) \mid v_0 \to v]\right], \tag{4}$$

where the expectation is over the non-significant vertices v in layer-i of part-j, reached by the path $\mathcal{T}$. (If $\mathcal{T}$ stops before reaching layer-i of part-j, or if it reaches a significant vertex, we think of v as undefined and think of the inner probability as 0). If $j = 1$, we are done by Claim 8. We will proceed with the case $j = 2$. Recall that $\ell_2 = \ell$.

We will use the following lemma, whose proof is deferred to the next subsection. We shall instantiate the lemma by setting $\mathcal{S}_v = \mathrm{Sig}(v)$.

▶ **Lemma 9.** *Assume that for every non-significant vertex v in layer-i of part-2, we have some subset of values $\mathcal{S}_v \subseteq X$ that depends only on v. Assume that for every such v (with positive probability for the event $v_1 \to v$, where v_1 is the vertex recorded by v), we have*

$$\Pr_x[x \in \mathcal{S}_v \mid v_1 \to v] \le 2^{-4\tilde{\ell}}.$$

Then,

$$\mathop{\boldsymbol{E}}_v[\Pr_x[x \in \mathcal{S}_v \mid v_0 \to v]] < 2^{-\Omega(\tilde{\ell})} \tag{5}$$

where the expectation is over the non-significant vertices v in layer-i of part-2, reached by the path $\mathcal{T}$. (If $\mathcal{T}$ stops before reaching layer-i of part-2, or if it reaches a significant vertex, we think of v as undefined and think of the inner probability as 0).

By Expression (3), the assumption of the lemma is satisfied by the choice $\mathcal{S}_v = \mathrm{Sig}(v)$. Thus, the conclusion of the lemma implies that

$$\mathop{\mathbf{E}}_v\left[\Pr_x[x \in \mathrm{Sig}(v) \mid v_0 \to v]\right] \le 2^{-\Omega(\tilde{\ell})}.$$

Thus, the probability that $\mathcal{T}$ stops because of stopping rule 2 is at most $2^{-\Omega(\tilde{\ell})}$, in each step, and taking a union bound over the length of the program, the probability that $\mathcal{T}$ stops because of stopping rule 2 is at most $2^{-\Omega(\tilde{\ell})}$.

5.3.3 Proof of Lemma 9

Proof. We could also write $\mathbf{E}_v[\Pr_x[x \in \mathcal{S}_v \mid v_0 \to v]]$ as

$$\sum_{v\in\mathcal{L}_{i,2}} \Pr[v_0 \to v]\cdot \Pr_x[x \in \mathcal{S}_v \mid v_0 \to v] = \sum_{v\in\mathcal{L}_{i,2}} \Pr[(x \in \mathcal{S}_v) \;\wedge\; (v_0 \to v)]$$

where $\mathcal{L}_{i,2}$ denotes the **non-significant** vertices v in layer-i of part-2, **that are reachable (with probability larger than 0) from the start vertex**.

Later on, we will define for every $v \in \mathcal{L}_{i,2}$, an event G_v that will occur with high probability. We will denote by $\bar{G_v}$, the complement of G_v. We will bound

$$\sum_{v \in \mathcal{L}_{i,2}} \Pr[(x \in \mathcal{S}_v) \ \wedge \ (v_0 \to v)],$$

by bounding separately

$$\sum_{v \in \mathcal{L}_{i,2}} \Pr[G_v \ \wedge \ (x \in \mathcal{S}_v) \ \wedge \ (v_0 \to v)] \tag{6}$$

and

$$\sum_{v \in \mathcal{L}_{i,2}} \Pr[\bar{G_v} \ \wedge \ (x \in \mathcal{S}_v) \ \wedge \ (v_0 \to v)] \tag{7}$$

The second expression will be bounded by

$$\sum_{v \in \mathcal{L}_{i,2}} \Pr[\bar{G_v} \ \wedge \ (v_0 \to v)], \tag{8}$$

that will be at most $2^{-\Omega(\tilde{\ell})}$ (see Claim 10). Thus, we will focus first on bounding Expression (6), which is equal to

$$\sum_{v \in \mathcal{L}_{i,2}} \sum_{x' \in \mathcal{S}_v} \Pr[G_v \ \wedge \ (x = x') \ \wedge \ (v_0 \to v)] \tag{9}$$

$$= \sum_{v \in \mathcal{L}_{i,2}} \sum_{x' \in \mathcal{S}_v} \Pr[G_v \ \wedge \ (v_0 \to v) \mid (x = x')] \cdot \Pr[x = x']. \tag{10}$$

Recall that for two vertices w_1, w_2 in the program, we denote by $w_1 \to w_2$ the event (over $x, a_1, \ldots, a_m$) that the path $\mathcal{T}$ that starts from w_1 reaches w_2.

Recall that by the construction of the branching-program $\hat{B}$, part-2 runs a copy of part-1 of the computation. Thus, the vertex v has a corresponding vertex v' in layer-i of part-1, such that, if the path $\mathcal{T}$ reached v it previously reached v'. Recall also that v remembers v_1, so if the path $\mathcal{T}$ reached v it previously reached v_1.

Thus, the event $v_0 \to v$ is equivalent to $v_0 \to v' \to v_1 \to v$, that is, the event

$$(v_0 \to v') \wedge (v' \to v_1) \wedge (v_1 \to v).$$

Since the second part of the program runs a copy of the first part, the event $v_1 \to v$ implies the event $v_0 \to v'$. Hence, the event $v_0 \to v$ is equivalent to the event

$$(v' \to v_1) \wedge (v_1 \to v).$$

Note also that if we fix x, that is, if we condition on $x = x'$, and we fix v (which also fixes v', v_1) the events $(v' \to v_1)$ and $(v_1 \to v)$ are independent (as the first one depends only on $a_{i+1}, \ldots, a_m$ and the second depends only on $a_1, \ldots, a_i$). We will also have the property that the event G_v is a function of v' rather than v, and hence will also be denoted by $G_{v'} = G_v$ (recall that v determines v'). Moreover, if we fix x and v', we will have the property that the event $G_{v'}$ depends only on $a_{i+1}, \ldots, a_m$, and hence the events $G_{v'}$ and $(v' \to v_1)$ are independent of $(v_1 \to v)$.

Thus, for a fixed v (which also fixes v', v_1) and any $x' \in X$,

$$\begin{aligned}\Pr[G_v \wedge (v_0 \to v) \mid x = x'] &= \Pr[G_{v'} \wedge (v' \to v_1) \wedge (v_1 \to v) \mid x = x'] \\ &= \Pr[G_{v'} \wedge (v' \to v_1) \mid x = x'] \cdot \Pr[v_1 \to v \mid x = x'].\end{aligned}$$

We introduce the event $(v' \widetilde{\to} v_1)$ to indicate that the computational path from v' reached v_1 (as opposed to the usual notation that denotes the truncated path). Since $(v' \to v_1)$ implies $(v' \widetilde{\to} v_1)$ we have

$$\Pr[G_{v'} \wedge (v' \to v_1) \mid x = x'] \cdot \Pr[v_1 \to v \mid x = x']$$

$$\leq \Pr[G_{v'} \wedge (v' \widetilde{\to} v_1) \mid x = x'] \cdot \Pr[v_1 \to v \mid x = x'].$$

By Bayes' rule, the last expression is at most

$$\Pr[x = x' \mid G_{v'} \wedge (v' \widetilde{\to} v_1)] \cdot \Pr[x = x' \mid v_1 \to v] \cdot \frac{\Pr[v' \widetilde{\to} v_1] \cdot \Pr[v_1 \to v]}{\Pr[x = x']^2}$$

$$= \mathbb{P}_{x|G_{v'} \wedge (v' \widetilde{\to} v_1)}(x') \cdot \mathbb{P}_{x|v_1 \to v}(x') \cdot \frac{\Pr[v' \widetilde{\to} v_1] \cdot \Pr[v_1 \to v]}{\Pr[x = x']^2}.$$

Thus, Expression (10) is at most

$$\sum_{v \in \mathcal{L}_{i,2}} \left(\Pr[v' \widetilde{\to} v_1] \cdot \Pr[v_1 \to v] \cdot \sum_{x' \in \mathcal{S}_v} \frac{\mathbb{P}_{x|G_{v'} \wedge (v' \widetilde{\to} v_1)}(x')}{\Pr[x = x']} \cdot \mathbb{P}_{x|v_1 \to v}(x') \right). \tag{11}$$

Note that from layer-i of part-1 to layer-m of part-1, the branching program is one-pass. Denote by $R_{v'}$ the one-pass branching program, from layer-i of part-1 to layer-m of part-1, with starting vertex v'. Thus, we can use what we already know about one-pass branching programs. We will apply a slight modification of the main theorem of [5] (Proposition 24 from Appendix), for one-pass branching programs, with parameters $k' = k, \ell' = \tilde{\ell}, r' = \tilde{r}/4$.

As $m \leq 2^{\epsilon \tilde{r}}$ and $R_{v'}$ has width at most $2^{\epsilon k \tilde{\ell}/2} \leq 2^{k' \cdot \ell'/100}$ (ϵ is small enough), by Proposition 24, we know that for any fixed v', there exists an event $G_{v'}$ that depends only on $x, a_{i+1}, \ldots, a_m$, such that, $\Pr(G_{v'}) \geq 1 - 2^{-\tilde{\ell}/8}$ ($\tilde{\ell} \leq k$), and for every $x' \in X$, and every v_1 such that $\Pr[G_{v'} \wedge (v' \widetilde{\to} v_1)] > 0$ it holds that

$$\mathbb{P}_{x|G_{v'} \wedge (v' \widetilde{\to} v_1)}(x') \leq 2^{2\tilde{\ell}} \cdot 2^{-n}.$$

Namely, the event $G_{v'}$ is the event G from Proposition 24 corresponding to the branching program $R_{v'}$ (that is, the event $G_{v'}$ is the event that the truncated-path as defined for one-pass branching programs in [5] with slight modification, didn't stop because of one of the stopping rules, until the last layer, and didn't violate the significant vertices and significant values stopping rules in the last layer, that is, layer-m of part-1).

Substituting this in Expression (11), we get that the expression is at most

$$2^{2\tilde{\ell}} \cdot \sum_{v \in \mathcal{L}_{i,2}} \left(\Pr[v' \widetilde{\to} v_1] \cdot \Pr[v_1 \to v] \cdot \sum_{x' \in \mathcal{S}_v} \mathbb{P}_{x|v_1 \to v}(x') \right). \tag{12}$$

By the assumption of the lemma, for any $v \in \mathcal{L}_{i,2}$ we have $\sum_{x' \in \mathcal{S}_v} \mathbb{P}_{x|v_1 \to v}(x') \leq 2^{-4\tilde{\ell}}$, thus Expression (12) is at most

$$2^{-2\tilde{\ell}} \cdot \sum_{v \in \mathcal{L}_{i,2}} \Pr[v' \widetilde{\to} v_1] \cdot \Pr[v_1 \to v].$$

Recall that $\mathcal{L}_{i,2}$ denotes only the vertices v in layer-i of part-2, that are reachable (with probability larger than 0) from the start vertex, v_0. Recall that the event $(v_1 \to v)$ is equivalent to the event $(v_0 \to v') \wedge (v_1 \to v)$.

Thus,

$$\begin{aligned}
\sum_{v \in \mathcal{L}_{i,2}} \Pr[v' \overset{\sim}{\to} v_1] \cdot \Pr[v_1 \to v] &\leq \sum_{v',v_1,v} \Pr\left[v' \overset{\sim}{\to} v_1\right] \cdot \Pr\left[(v_0 \to v') \wedge (v_1 \to v)\right] \\
&= \sum_{v',v_1} \Pr\left[v' \overset{\sim}{\to} v_1\right] \cdot \left(\sum_{v} \Pr\left[(v_0 \to v') \wedge (v_1 \to v)\right] \right) \\
&\leq \sum_{v',v_1} \Pr\left[v' \overset{\sim}{\to} v_1\right] \cdot \Pr\left[v_0 \to v'\right] \\
&= \sum_{v'} \Pr\left[v_0 \to v'\right] \cdot \left(\sum_{v_1} \Pr\left[v' \overset{\sim}{\to} v_1\right] \right) \\
&\leq \sum_{v'} \Pr\left[v_0 \to v'\right] \leq 1
\end{aligned}$$

(where the possible inequality in the first line is because the first sum is on all the paths $v_0 \to v' \to v_1 \to v$, obtained with positive probabilities, whereas the second sum is on all possible vertices v_0, v', v_1, v in the corresponding layers of the branching program).

Thus, we conclude that Expression (6) is at most $2^{-2\tilde{\ell}}$. It remains to bound Expression (8).

▷ Claim 10.

$$\sum_{v \in \mathcal{L}_{i,2}} \Pr[\bar{G_v} \wedge (v_0 \to v)] \leq 2^{-\Omega(\tilde{\ell})}.$$

Proof.

$$\begin{aligned}
\sum_{v \in \mathcal{L}_{i,2}} \Pr[\bar{G_v} \wedge (v_0 \to v)] &= \sum_{v \in \mathcal{L}_{i,2}} \Pr[\bar{G_v} \wedge (v_0 \to v') \wedge (v' \to v_1) \wedge (v_1 \to v)] \\
&\leq \sum_{v',v_1,v} \Pr[\bar{G_v} \wedge (v_0 \to v') \wedge (v' \to v_1) \wedge (v_1 \to v)] \\
&= \sum_{v',v_1,v} \Pr[\bar{G_{v'}} \wedge (v_0 \to v') \wedge (v' \to v_1) \wedge (v_1 \to v)] \\
= \sum_{v' \in \mathcal{L}_{i,1}} \Pr[\bar{G_{v'}} \wedge (v_0 \to v')] \cdot \sum_{v_1,v} &\Pr[(v' \to v_1) \wedge (v_1 \to v) | \bar{G_{v'}} \wedge (v_0 \to v')] \\
&\leq \sum_{v' \in \mathcal{L}_{i,1}} \Pr[\bar{G_{v'}} \wedge (v_0 \to v')]. \qquad (13)
\end{aligned}$$

For every non-significant $v' \in \mathcal{L}_{i,1}$, denote by

$$\mathcal{X}_{v'} = \{x' : \mathbb{P}_{x|v_0 \to v'}(x') \geq 2^{\tilde{\ell}/16} \cdot 2^{-n}\},$$

and split the expression $\Pr[\bar{G_{v'}} \wedge (v_0 \to v')]$ according to whether or not $(x \in \mathcal{X}_{v'})$.

$$\Pr[\bar{G_{v'}} \wedge (v_0 \to v')] \leq \Pr[(v_0 \to v') \wedge (x \in \mathcal{X}_{v'})] + \Pr[\bar{G_{v'}} \wedge (v_0 \to v') \wedge (x \notin \mathcal{X}_{v'})] \qquad (14)$$

We begin by bounding the first summand in Expression (14):

$$\Pr[(v_0 \to v') \wedge (x \in \mathcal{X}_{v'})] = \Pr(v_0 \to v') \cdot \Pr[(x \in \mathcal{X}_{v'}) | v_0 \to v']$$

We bound $\Pr[x \in \mathcal{X}_{v'} | v_0 \to v']$ very similarly to the proof of Claim 8, but with a different threshold. Since v' is not significant,

$$\mathop{\mathbf{E}}_{x' \sim \mathbb{P}_{x|v_0 \to v'}} \left[\mathbb{P}_{x|v_0 \to v'}(x')\right] = \sum_{x' \in X} \left[\mathbb{P}_{x|v_0 \to v'}(x')^2\right] = 2^n \cdot \mathop{\mathbf{E}}_{x' \in_R X} \left[\mathbb{P}_{x|v_0 \to v'}(x')^2\right] \leq 2^{2\ell_1} \cdot 2^{-n}.$$

Hence, by Markov's inequality,

$$\Pr[x \in \mathcal{X}_{v'} | v_0 \to v'] = \Pr_{x' \sim \mathbb{P}_{x|v_0 \to v'}} \left[\mathbb{P}_{x|v_0 \to v'}(x') \geq 2^{\tilde{\ell}/16} \cdot 2^{-n}\right] \leq 2^{2\ell_1 - \tilde{\ell}/16} \leq 2^{-\tilde{\ell}/32}$$

(recall that $\ell_1 = \tilde{\ell}/100$). Overall, we bounded the first summand in Expression (14) by $\Pr(v_0 \to v') \cdot 2^{-\tilde{\ell}/32}$.

Next, we bound the second summand in Expression (14).

$$\Pr\left[\bar{G_{v'}} \wedge (v_0 \to v') \wedge (x \notin \mathcal{X}_{v'})\right]$$

$$= \sum_{x' \in X \setminus \mathcal{X}_{v'}} \Pr(x = x') \cdot \Pr\left[\bar{G_{v'}} \wedge (v_0 \to v') \mid x = x'\right].$$

Since if we fix x and v', the event $G_{v'}$ depends only on $a_{i+1}, \ldots, a_m$ and hence is independent of $(v_0 \to v')$, we have

$$\sum_{x' \in X \setminus \mathcal{X}_{v'}} \Pr(x = x') \cdot \Pr\left[\bar{G_{v'}} \wedge (v_0 \to v') \mid x = x'\right]$$

$$= \sum_{x' \in X \setminus \mathcal{X}_{v'}} \Pr(x = x') \cdot \Pr\left[\bar{G_{v'}} \mid x = x'\right] \cdot \Pr\left[v_0 \to v' \mid x = x'\right]$$

$$= \Pr\left(v_0 \to v'\right) \cdot \sum_{x' \in X \setminus \mathcal{X}_{v'}} \Pr\left[\bar{G_{v'}} \mid x = x'\right] \cdot \Pr\left[x = x' \mid v_0 \to v'\right]$$

(by Bayes' rule)

$$= \Pr\left(v_0 \to v'\right) \cdot \sum_{x' \in X \setminus \mathcal{X}_{v'}} \Pr\left[\bar{G_{v'}} \mid x = x'\right] \cdot \mathbb{P}_{x|v_0 \to v'}(x')$$

$$\leq \Pr\left(v_0 \to v'\right) \cdot \sum_{x' \in X \setminus \mathcal{X}_{v'}} \Pr\left[\bar{G_{v'}} \mid x = x'\right] \cdot 2^{\tilde{\ell}/16} \cdot 2^{-n}$$

(by the definition of $\mathcal{X}_{v'}$)

$$\leq \Pr\left(v_0 \to v'\right) \cdot \Pr\left(\bar{G_{v'}}\right) \cdot 2^{\tilde{\ell}/16}$$

$$\leq \Pr\left(v_0 \to v'\right) \cdot 2^{-\tilde{\ell}/8} \cdot 2^{\tilde{\ell}/16}$$

$$\leq \Pr\left(v_0 \to v'\right) \cdot 2^{-\tilde{\ell}/16}.$$

Substituting in Expression (14), we have

$$\Pr[\bar{G_{v'}} \wedge (v_0 \to v')] \leq \Pr\left(v_0 \to v'\right) \cdot 2^{-\tilde{\ell}/32} + \Pr\left(v_0 \to v'\right) \cdot 2^{-\tilde{\ell}/16}.$$

Substituting in Expression (13), we have

$$\sum_{v \in \mathcal{L}_{i,2}} \Pr[\bar{G_v} \wedge (v_0 \to v)] \leq (2^{-\tilde{\ell}/32} + 2^{-\tilde{\ell}/16}) \cdot \sum_{v' \in \mathcal{L}_{i,1}} \Pr\left(v_0 \to v'\right) \leq 2 \cdot 2^{-\tilde{\ell}/32}. \quad \blacktriangleleft$$

This finishes the proof of Lemma 9. $\blacktriangleleft$

5.3.4 Stopping Rule 3: Bad-Edges

We will now bound the probability that $\mathcal{T}$ stops because of stopping rule 3. We will first prove the following claim.

▷ Claim 11. If v is a non-significant vertex in layer-i of part-j (where $j \in \{1,2\}$), then

$$\Pr_{a_{i+1}}[a_{i+1} \in \mathrm{Bad}(v)] \leq 2^{-4k}.$$

Proof. Since v is not significant, $\left\|\mathbb{P}_{x|v_{j-1}\to v}\right\|_2 \leq 2^{\ell_j} \cdot 2^{-n} \leq 2^{\ell} \cdot 2^{-n}$. Since $\mathbb{P}_{x|v_{j-1}\to v}$ is a distribution, $\left\|\mathbb{P}_{x|v_{j-1}\to v}\right\|_1 = 2^{-n}$. Thus,

$$\frac{\left\|\mathbb{P}_{x|v_{j-1}\to v}\right\|_2}{\left\|\mathbb{P}_{x|v_{j-1}\to v}\right\|_1} \leq 2^{\ell}.$$

Since M is a $(10k, 10\ell)$-L_2-extractor with error 2^{-10r}, there are at most $2^{-10k} \cdot |A|$ elements $a \in A$ with

$$\left|\langle M_a, \mathbb{P}_{x|v_{j-1}\to v}\rangle\right| \geq 2^{-10r} \cdot \left\|\mathbb{P}_{x|v_{j-1}\to v}\right\|_1 = 2^{-10r} \cdot 2^{-n}$$

The claim follows since a_{i+1} is uniformly distributed over A. ◁

By Claim 11, if v is a non-significant vertex then

$$\Pr_{a_{i+1}}[a_{i+1} \in \mathrm{Bad}(v)] \leq 2^{-4k}.$$

We need to bound

$$\Pr_{a_{i+1}}[a_{i+1} \in \mathrm{Bad}(v) \setminus \mathrm{High}(v) \mid v_0 \to v].$$

We bound

$$\begin{aligned}\Pr_{a_{i+1}}[a_{i+1} \in \mathrm{Bad}(v) \setminus \mathrm{High}(v) \mid v_0 \to v] &= \sum_{a \in \mathrm{Bad}(v)\setminus\mathrm{High}(v)} \Pr[a_{i+1} = a \mid v_0 \to v] \\ &\leq \sum_{a \in \mathrm{Bad}(v)\setminus\mathrm{High}(v)} 2^k \cdot 2^{-n'} \leq 2^k \cdot \sum_{a \in \mathrm{Bad}(v)} \Pr[a_{i+1} = a] \\ &= 2^k \cdot \Pr_{a_{i+1}}[a_{i+1} \in \mathrm{Bad}(v)] \leq 2^k \cdot 2^{-4k} = 2^{-3k}.\end{aligned}$$

Thus, the probability that $\mathcal{T}$ stops because of stopping rule 3 is at most 2^{-3k}, in each step, and taking a union bound over the length of the program, the probability that $\mathcal{T}$ stops because of stopping rule 3 is at most 2^{-2k}.

5.3.5 Stopping Rule 4: Very-Bad Edges

We will now bound the probability that $\mathcal{T}$ stops because of stopping rule 4.

Recall that for a vertex v in layer-i of part-2 of the program, $\mathrm{VeryBad}(v)$ is the set of all $(a,b) \in A \times \{-1,1\}$, such that,

$$\Pr_x[M(a,x) = b \mid v_1 \to v] \leq 2^{-4\tilde{\ell}}.$$

Note that for every $a \in A$, there is at most one $b \in \{-1,1\}$, denoted $b_v(a)$, such that

$$\Pr_x[M(a,x) = b \mid v_1 \to v] \leq 2^{-4\tilde{\ell}}.$$

If such a b doesn't exist we let $b_v(a) = *$, and think of it as undefined. Thus, for every v, and every $(a,b) \in A \times \{-1,1\}$,

$$\big((a,b) \in \text{VeryBad}(v)\big) \iff \big(b = b_v(a)\big), \tag{15}$$

and

$$\Pr_x[M(a,x) = b_v(a) \mid v_1 \to v] \leq 2^{-4\tilde{\ell}}. \tag{16}$$

Let $a_v \in A$ be an $a \in A$, such that $\Pr_x[M(a,x) = b_v(a) \mid v_0 \to v]$ is maximal and let $b_v = b_v(a_v)$. We need to bound from above

$$\mathop{\mathbf{E}}_v \big[\Pr[(a_{i+1}, b_{i+1}) \in \text{VeryBad}(v) \mid v_0 \to v]\big], \tag{17}$$

where the expectation is over the vertex v in layer-i of part-2, reached by the path $\mathcal{T}$. (If $\mathcal{T}$ stops before reaching layer-i of part-2, we think of v as undefined and think of the inner probability as 0). That is, we could also write Expression (17) as

$$\sum_{v \in \mathcal{L}_{i,2}} \Pr[v_0 \to v] \cdot \Pr[(a_{i+1}, b_{i+1}) \in \text{VeryBad}(v) \mid v_0 \to v],$$

where $\mathcal{L}_{i,2}$ denotes the vertices v in layer-i of part-2, **that are reachable (with probability larger than 0) from the start vertex**. By Equation (15), Expression (17) is equal to

$$\mathop{\mathbf{E}}_v \big[\Pr[b_{i+1} = b_v(a_{i+1}) \mid v_0 \to v]\big],$$

which, by the definition of b_{i+1}, is equal to

$$\mathop{\mathbf{E}}_v \big[\Pr[M(a_{i+1}, x) = b_v(a_{i+1}) \mid v_0 \to v]\big],$$

which, by the definitions of a_v, b_v, is at most

$$\mathop{\mathbf{E}}_v \big[\Pr[M(a_v, x) = b_v \mid v_0 \to v]\big]. \tag{18}$$

In what follows, we assume for simplicity and without loss of generality that for every v, $b_v \in \{-1,1\}$ is defined (as otherwise $\Pr[M(a_v,x) = b_v \mid v_0 \to v] = 0$ and can be omitted from the expectation).

For any fixed v, denote by $\mathcal{S}_v = \{x : M(a_v,x) = b_v\}$. We can apply Lemma 9, since from Expression (16) for any non-significant v

$$\Pr[x \in \mathcal{S}_v \mid v_1 \to v] \leq 2^{-4\tilde{\ell}}.$$

Thus, we get

$$\mathop{\mathbf{E}}_v[\Pr[x \in \mathcal{S}_v \mid v_0 \to v]] \leq 2^{-\Omega(\tilde{\ell})},$$

and since $(x \in \mathcal{S}_v) \iff (M(a_v,x) = b_v)$, we have

$$\mathop{\mathbf{E}}_v[\Pr[M(a_v,x) = b_v \mid v_0 \to v]] \leq 2^{-\Omega(\tilde{\ell})}.$$

Finally, by the definitions of a_v and b_v we have

$$\mathop{\mathbf{E}}_v[\Pr[(a_{i+1}, b_{i+1}) \in \text{VeryBad}(v) \mid v_0 \to v]] \leq \mathop{\mathbf{E}}_v[\Pr[M(a_v,x) = b_v \mid v_0 \to v]] \leq 2^{-\Omega(\tilde{\ell})}.$$

Thus, the probability that $\mathcal{T}$ stops because of stopping rule 4 is at most $2^{-\Omega(\tilde{\ell})}$, in each step, and taking a union bound over the length of the program, the probability that $\mathcal{T}$ stops because of stopping rule 4 is at most $2^{-\Omega(\tilde{\ell})}$.

5.3.6 Stopping Rule 5: Large L_v

Recall that for two vertices w_1, w_2 in the program, we denote by $w_1 \to w_2$ the event (over $x, a_1, \ldots, a_m$) that the path $\mathcal{T}$ that starts from w_1 reaches w_2.

Recall that v_1 is the vertex reached by the path at the end of part-1. Fix v_1 and denote by E the event $v_0 \to v_1$. Let $u_0, u_1, \ldots, u_m$ be the vertices reached by the path in part-2, where $u_0 = v_1$. (If the path stops before reaching layer-i of part-2, we define u_i to be a special *stop* vertex in that layer). Note that conditioned on the event E, the random variable u_i is a function of $x, a_1, \ldots, a_i$ and for $i \geq 1$ it can also be viewed as a function of x, u_{i-1}, a_i.

Denote by T the number of high-probability edges that the path traverses in part-2. For every $i \in [m]$, let $T_i \in \{0,1\}$ be an indicator random variable that indicates whether the path traverses a high-probability edge at step-i of part-2. Thus,

$$T = \sum_{i=1}^{m} T_i.$$

For every $i \in [m]$, we have that $T_i = 1$ only if $a_i \in \mathrm{High}(u_{i-1})$, that is, only if $\Pr(a_i | u_{i-1}, E) \geq 2^k \cdot 2^{-n'}$, or equivalently

$$\frac{\log\left(2^{n'} \cdot \Pr(a_i | u_{i-1}, E)\right)}{k} \geq 1.$$

▷ Claim 12. Let $Z \in \{0,1\}^{n'}$ be any random variable. Let $k \geq 4$. Let $T(Z) \in \{0,1\}$ be an indicator random variable for the event $\Pr(Z) \geq 2^k \cdot 2^{-n'}$. Then,

$$2 \cdot \mathop{\mathbf{E}}_{Z}\left[\frac{\log\left(2^{n'} \cdot \Pr(Z)\right)}{k}\right] \geq \mathop{\mathbf{E}}_{Z}[T(Z)].$$

Proof. Let $\alpha = \Pr_Z(T(Z) = 1)$. That is, we have $\Pr(Z) \geq 2^k \cdot 2^{-n'}$ with probability α. Thus,

$$\mathop{\mathbf{E}}_{Z}\left[\frac{\log\left(2^{n'} \cdot \Pr(Z)\right)}{k}\right] =$$

$$\alpha \cdot \mathop{\mathbf{E}}_{Z}\left[\frac{\log\left(2^{n'} \cdot \Pr(Z)\right)}{k} \,\middle|\, T(Z) = 1\right] + (1-\alpha) \cdot \mathop{\mathbf{E}}_{Z}\left[\frac{\log\left(2^{n'} \cdot \Pr(Z)\right)}{k} \,\middle|\, T(Z) = 0\right].$$

By the monotonicity of the logarithm function, we have,

$$\alpha \cdot \mathop{\mathbf{E}}_{Z}\left[\frac{\log\left(2^{n'} \cdot \Pr(Z)\right)}{k} \,\middle|\, T(Z) = 1\right] \geq \alpha \cdot \mathop{\mathbf{E}}_{Z}\left[\frac{\log\left(2^{n'} \cdot 2^k \cdot 2^{-n'}\right)}{k} \,\middle|\, T(Z) = 1\right] = \alpha$$

By the monotonicity of the logarithm function and the concavity of the entropy function, we have,

$$(1-\alpha) \cdot \mathop{\mathbf{E}}_{Z}\left[\frac{\log\left(2^{n'} \cdot \Pr(Z)\right)}{k} \,\middle|\, T(Z) = 0\right] \geq$$

$$(1-\alpha) \cdot \mathop{\mathbf{E}}_{Z}\left[\frac{\log\left(2^{n'} \cdot (1-\alpha) \cdot 2^{-n'}\right)}{k} \,\middle|\, T(Z) = 0\right] =$$

$$\frac{(1-\alpha)\log(1-\alpha)}{k}$$

(as, by the concavity of the entropy function, the expression is minimized when the random variable $Z|(T(Z)=0)$ is uniformly distributed).

Thus, the left hand side of the claim is at least

$$2\alpha + \frac{2(1-\alpha)\log(1-\alpha)}{k} \geq 2\alpha - \frac{4\alpha}{k} \geq 2\alpha - \frac{4\alpha}{4} = \alpha.$$

The claim follows Since $\mathbf{E}_Z[T(Z)] = \alpha$. ◁

By Claim 12,

$$\mathop{\mathbf{E}}_{x,a_1,\ldots,a_m}[T|E] = \sum_{i=1}^{m} \mathop{\mathbf{E}}_{x,a_1,\ldots,a_m}[T_i|E] \leq 2 \cdot \sum_{i=1}^{m} \mathop{\mathbf{E}}_{x,a_1,\ldots,a_m}\left[\frac{\log\left(2^{n'} \cdot \Pr(a_i|u_{i-1},E)\right)}{k}\right]$$

$$= \frac{2}{k} \cdot \left(mn' - \sum_{i=1}^{m} \mathbf{H}(a_i|u_{i-1},E)\right),$$

where $\mathbf{H}$ denotes the entropy function. Since conditioning may only decrease the entropy, the last expression is at most

$$\leq \frac{2}{k} \cdot \left(mn' - \sum_{i=1}^{m} \mathbf{H}(a_i|x,u_{i-1},E)\right).$$

Since, conditioned on E, the random variable u_{i-1} is a function of $x, a_1, \ldots, a_{i-1}$, by the data-processing inequality, $\mathbf{H}(a_i|x,u_{i-1},E) \geq \mathbf{H}(a_i|x,a_1,\ldots,a_{i-1},E)$, and hence the last expression is at most

$$\leq \frac{2}{k} \cdot \left(mn' - \sum_{i=1}^{m} \mathbf{H}(a_i|x,a_1,\ldots,a_{i-1},E)\right).$$

By the chain rule, the last expression is equal to

$$= \frac{2}{k} \cdot \left(mn' - \mathbf{H}(a_1,\ldots,a_m|x,E)\right)$$

$$= \frac{2}{k} \cdot \left(mn' - \mathbf{H}(x,a_1,\ldots,a_m|E) + \mathbf{H}(x|E)\right)$$

$$\leq \frac{2}{k} \cdot \left(mn' + n - \mathbf{H}(x,a_1,\ldots,a_m|E)\right)$$

$$\leq \frac{2}{k} \cdot \log\left(\tfrac{1}{\Pr(E)}\right).$$

Thus,

$$\mathop{\mathbf{E}}_{x,a_1,\ldots,a_m}[T|E] \leq \frac{2}{k} \cdot \log\left(\tfrac{1}{\Pr(E)}\right).$$

By Markov inequality

$$\Pr_{x,a_1,\ldots,a_m}\left[T \geq \frac{200}{k} \cdot \log\left(\tfrac{1}{\Pr(E)}\right) \middle| E\right] \leq \frac{1}{100}.$$

Since we assumed that $\Pr(E) \geq 2^{-10\tilde{r}} \cdot d_1^{-1}$ and since the width of $\hat{B}$ is at most $2^{\epsilon k \tilde{\ell}/2}$ and since by Equation (1) and Equation (2), $\tilde{r}$ is negligible compared to $\epsilon k\tilde{\ell}/2$, we have that $\log\left(\frac{1}{\Pr(E)}\right) \leq \epsilon k \tilde{\ell}$. Hence,

$$\Pr_{x,a_1,\ldots,a_m}\left[T \geq 200\epsilon\tilde{\ell} \;\middle|\; E\right] \leq \frac{1}{100}.$$

Thus, the probability to stop on part-2 because of stopping rule 5 is at most $\frac{1}{100}$.

5.3.7 Stopping Rule 6: Consistency-Stop

We will now show that the probability that $\mathcal{T}$ stops on a vertex v, in layer-i of part-2, because of stopping rule 6, conditioned on the event $v_0 \to v$, is 0.

Recall that by the construction of the branching-program $\hat{B}$, part-2 runs a copy of part-1 of the computation. Thus, the vertex v has a corresponding vertex v' in layer-i of part-1, such that, if the path $\mathcal{T}$ reached v it previously reached v'.

If $\mathcal{T}$ needs to stop on v, because of stopping rule 6, because $\mathcal{T}$ stopped on the vertex v', it couldn't have reached v in the first place (as it would have stopped on v'). Thus, conditioned on the event $v_0 \to v$, the path $\mathcal{T}$ didn't stop on v' and doesn't need to stop on v because of stopping rule 6.

Thus, the probability that $\mathcal{T}$ stops because of stopping rule 6 is 0.

This completes the proof of Lemma 6. ◀

5.4 The Final Success Probability is Small

Let v be a vertex in the last layer of the program. Assume that the probability for the event $v_0 \to v$ is larger than 0. Since v is in the last layer, the event $v_0 \to v$ is equivalent to $v_1 \to v$ (since the second part of the program runs a copy of the first part). Hence,

$$\mathbb{P}_{x|v_0\to v} = \mathbb{P}_{x|v_1\to v}$$

and

$$\Pr[v_0 \to v] = \Pr[v_1 \to v].$$

In particular, if v is not significant, $\mathbb{P}_{x|v_0\to v}$ has small L_2-norm.

$$\mathop{\mathbf{E}}_{x' \in_R X}\left[\mathbb{P}_{x|v_0\to v}(x')^2\right] \leq 2^{2\ell} \cdot 2^{-2n}.$$

Hence, for every $x' \in X$,

$$\Pr[x = x' \mid v_0 \to v] = \mathbb{P}_{x|v_0\to v}(x') \leq 2^{\ell} \cdot 2^{-n/2} \leq 2^{-n/4}$$

In particular,

$$\Pr[\tilde{x}(v) = x \mid v_0 \to v] \leq 2^{-n/4}.$$

Thus, either the computation path stops before reaching v which happens with probability at most $\frac{1}{100} + o(1)$ or it reaches a non-significant vertex where the probability of guessing correctly is $o(1)$. Thus, the final success probability is bounded by $\frac{1}{100} + o(1)$. This completes the proof of Theorem 4.

6 Proof of Lemma 7

Proof Overview

Let s be a significant vertex in part-j (that remembers the vertices visited at the end of parts $1,\ldots,j-1$, denoted by $s_1,\ldots,s_{j-1}$). Assume that the probability for the event $v_0 \to s$ is larger than 0. We need to bound from above the probability for the event $v_0 \to s$. Since the event $v_0 \to s$ is equivalent to $(v_0 \to s_{j-1}) \wedge (s_{j-1} \to s)$, it suffices to bound from above the probability for $(s_{j-1} \to s)$. Note that to analyze this probability we can ignore all parts of the program, except for part-j, which is a one-pass branching program.

We would like to reprove Lemma 4.1 of [5], with the updated stopping rules. In the definition of the progress function $\mathcal{Z}_i$, we will take the sum only on vertices $u \in \mathcal{L}_{i,j}$, such that s can be reached from u (and in the same way for edges in the definition of $\mathcal{Z}'_i$). In particular, this implies that every index in L_u is contained in L_s (as otherwise s cannot be reached from u).

The progress function is still small at the beginning and large at the end, so as before the main thing to do is to prove that it grows slowly. This was done in Claim 4.10 of [5].

The main difference here is that the progress function doesn't grow slowly for every edge, as some edges are now bad, and we have to take the bad edges into account. We separate to time steps that are in L_s and time steps that are not in L_s. For time steps that are not in L_s, we don't need to count the bad edges at all, as they are not recorded by L_s and hence s is not reachable from these edges.

As for steps in L_s, we know that the edges are not very-bad, and we show that the progress function may increase by a factor of at most $2^{5\tilde{\ell}k}$. Since $|L_s| \leq 200\epsilon\tilde{\ell}$ (as otherwise $\mathcal{T}$ would have stopped by stopping rule 5), the total effect of the bad edges on the progress function is a factor of at most $2^{5\tilde{\ell}k \cdot 200\epsilon\tilde{\ell}} \leq 2^{1000\epsilon k\ell}$, which we can afford.

6.1 Proof of Lemma 7

Proof. We need to prove that the probability that $\mathcal{T}$ reaches any significant vertex is $o(1)$. Let s be a significant vertex in part-j. Assume that the probability that $\mathcal{T}$ reaches s is larger than 0. We will bound from above the probability that $\mathcal{T}$ reaches s, and then use a union bound over all significant vertices of $\hat{B}$. Since the event $v_0 \to s$ is equivalent to $(v_0 \to s_{j-1}) \wedge (s_{j-1} \to s)$, it suffices to bound from above the probability for $(s_{j-1} \to s)$. Note that to analyze this probability we can ignore all parts other than j of the program, which leaves us with a one-pass branching program. Furthermore, since s determines s_{j-1}, we can only consider the subprogram that starts at s_{j-1} and analyze the probability that the restriction of $\mathcal{T}$ to this subprogram reaches s. We denote by B' the subprogram of $\hat{B}$ restricted to the j-part with s_{j-1} as the starting node.

The Distributions $\mathbb{P}_{x|v}$ and $\mathbb{P}_{x|e}$

For a vertex v in B', we denote by E_v the event that $\mathcal{T}$ starting from s_{j-1} reaches the vertex v. For simplicity, we denote by $\Pr(v) = \Pr(E_v)$ the probability for E_v (where the probability is over $x, a_1, \ldots, a_m$), and we denote by $\mathbb{P}_{x|v} = \mathbb{P}_{x|E_v}$ the distribution of the random variable x conditioned on the event E_v.

Similarly, for an edge e of the branching program B', let E_e be the event that $\mathcal{T}$ starting from s_{j-1} traverses the edge e. Denote, $\Pr(e) = \Pr(E_e)$ (where the probability is over $x, a_1, \ldots, a_m$), and $\mathbb{P}_{x|e} = \mathbb{P}_{x|E_e}$.

Notation

B' inherits the definitions of significant vertices, $\mathrm{Sig}(v)$, $\mathrm{Bad}(v)$, $\mathrm{VeryBad}(v)$ and $\mathrm{High}(v)$ from $\hat{B}$. Note that significant vertices, $\mathrm{Sig}(v)$, $\mathrm{Bad}(v)$ and $\mathrm{VeryBad}(v)$ are defined conditioned on the event $v_{j-1} \to v$, which is equivalent to the event E_v. Recall that the walk $\mathcal{T}$ does not stop on an edge (v,u) marked (a,b) if $a \in \mathrm{High}(v)$, as long as $(a,b) \notin \mathrm{VeryBad}(v)$. We will use the following fact on $\mathcal{T}$: if $i \notin L_s$ and $\mathcal{T}$ takes a bad-edge (v,u) on the i-th step, then $L_u \not\subseteq L_s$ and s is not reachable from u.

For $i \in \{0,\ldots,m\}$, let $\mathcal{L}'_i$ be the set of vertices v in layer-i of B' such that $\Pr(v) > 0$ and **it is possible to reach s from v** (in particular, the set of high-probability equations stored in v is also stored in s). For $i \in \{1,\ldots,m\}$, let Γ_i be the set of edges e from $\mathcal{L}'_{i-1}$ to $\mathcal{L}'_i$ of B', such that $\Pr(e) > 0$.

Recall that by the construction of the branching-program $\hat{B}$, part-j runs a copy of all previous parts of the computation. Thus, a vertex v in B' or equivalently a vertex v in part-j of $\hat{B}$ has corresponding vertices $v'_1,\ldots,v'_{j-1}$ in layer-i of parts $1,\ldots,j-1$, respectively, such that, if the path $\mathcal{T}$ reached v it previously reached $v'_1,\ldots,v'_{j-1}$. We denote by $v'_j = v$. We denote by

$$\widetilde{\mathrm{Sig}}(v) \triangleq \bigcup_{j'=1}^{j} \mathrm{Sig}(v'_{j'}).$$

Recall that by stopping rules 2 and 6, the path $\mathcal{T}$ stops if $x \in \widetilde{\mathrm{Sig}}(v)$.

The next claim bounds the probability of stopping on a vertex v in part-2 due to stopping rule 2 of part-1 on the vertex v' that v remembers.

▷ Claim 13. If v is a non-significant vertex in layer-i of part-2 that remembers v', and v' is a non-significant vertex in layer-i of part-1, then

$$\Pr_x[x \in \mathrm{Sig}(v') \mid v_1 \to v] \le 2^{-2\ell}.$$

Proof. Since v is not significant,

$$\mathop{\mathbf{E}}_{x' \sim \mathbb{P}_{x|v_1 \to v}} \left[\mathbb{P}_{x|v_0 \to v'}(x')\right] = \sum_{x' \in X} \left[\mathbb{P}_{x|v_0 \to v'}(x') \cdot \mathbb{P}_{x|v_1 \to v}(x')\right]$$

(using Cauchy-Schwarz)

$$\le \sqrt{\sum_{x' \in X} \mathbb{P}_{x|v_0 \to v'}(x')^2 \cdot \sum_{x' \in X} \mathbb{P}_{x|v_1 \to v}(x')^2}$$

$$= 2^n \cdot \sqrt{\mathop{\mathbf{E}}_{x' \in_R X} \left[\mathbb{P}_{x|v_0 \to v'}(x')^2\right] \mathop{\mathbf{E}}_{x' \in_R X} \left[\mathbb{P}_{x|v_1 \to v}(x')^2\right]}$$

(since both v' and v are non-significant)

$$\le 2^{\ell_1+\ell_2} \cdot 2^{-n}.$$

Hence, by Markov's inequality,

$$\Pr_{x' \sim \mathbb{P}_{x|v_1 \to v}} \left[\mathbb{P}_{x|v_0 \to v'}(x') > 2^{4\ell} \cdot 2^{-n}\right] \le 2^{\ell_1+\ell_2-4\ell} \le 2^{-2\ell}.$$

Since conditioned on the event $v_1 \to v$, the distribution of x is $\mathbb{P}_{x|v_1 \to v}$, we obtain

$$\Pr_x \left[x \in \mathrm{Sig}(v') \;\middle|\; v_1 \to v\right] = \Pr_x \left[\left(\mathbb{P}_{x|v_0 \to v}(x) > 2^{4\ell} \cdot 2^{-n}\right) \;\middle|\; v_1 \to v\right] \le 2^{-2\ell}. \quad \blacktriangleleft$$

▷ Claim 14. Let $i \in \{1,\ldots,m\}$. For any edge $e=(v,u) \in \Gamma_i$, labeled by (a,b), such that $\Pr(e) > 0$, for any $x' \in X$,

$$\mathbb{P}_{x|e}(x') = \begin{cases} 0 & \text{if} \quad x' \in \widetilde{\mathrm{Sig}}(v) \quad \text{or} \quad M(a,x') \neq b \\ \mathbb{P}_{x|v}(x') \cdot c_e^{-1} & \text{if} \quad x' \notin \widetilde{\mathrm{Sig}}(v) \quad \text{and} \quad M(a,x') = b \end{cases}$$

where c_e is a normalization factor that satisfies

- $c_e \geq \frac{1}{2} - 2 \cdot 2^{-2r}$, if $i \notin L_s$.
- $c_e \geq 2^{-4\tilde{\ell}} - 2 \cdot 2^{-2\ell} \geq 2^{-5\tilde{\ell}}$, if $i \in L_s$.

Proof. Let $v'_1,\ldots,v'_j$ be the vertices in the branching program $\hat{B}$ that v remembers. Let $e=(v,u)$ be an edge of B', labeled by (a,b), and such that $\Pr(e) > 0$. Since $\Pr(e) > 0$, the vertices $v'_1,\ldots,v'_j$ are not significant (as otherwise $\mathcal{T}$ always stops on v and hence $\Pr(e)=0$). Also, since $\Pr(e) > 0$, we know that (a,b) is not very-bad (as otherwise $\mathcal{T}$ never traverses e and hence $\Pr(e)=0$).

If $\mathcal{T}$ reaches v, it traverses the edge e if and only if: $x \notin \widetilde{\mathrm{Sig}}(v)$ (as otherwise $\mathcal{T}$ stops on v) and $M(a,x)=b$ and $a_{i+1}=a$. Therefore, for any $x' \in X$,

$$\mathbb{P}_{x|e}(x') = \begin{cases} 0 & \text{if} \quad x' \in \widetilde{\mathrm{Sig}}(v) \quad \text{or} \quad M(a,x') \neq b \\ \mathbb{P}_{x|v}(x') \cdot c_e^{-1} & \text{if} \quad x' \notin \widetilde{\mathrm{Sig}}(v) \quad \text{and} \quad M(a,x') = b \end{cases}$$

where c_e is a normalization factor, given by

$$c_e = \sum_{\left\{x' \,:\, x' \notin \widetilde{\mathrm{Sig}}(v) \,\wedge\, M(a,x')=b\right\}} \mathbb{P}_{x|v}(x') \;=\; \Pr_x[(x \notin \widetilde{\mathrm{Sig}}(v)) \wedge (M(a,x)=b) \mid E_v].$$

Since $v'_1,\ldots,v'_j$ are not significant, by Claim 8 and Claim 13:

$$\Pr_x[x \in \widetilde{\mathrm{Sig}}(v) \mid E_v] \leq \sum_{j'=1}^{j} 2^{-2\ell} \leq 2 \cdot 2^{-2\ell} \leq 2^{-2r}.$$

If $i \notin L_s$, then $a \notin \mathrm{Bad}(v)$, as otherwise $L_u \not\subseteq L_s$ and s is not reachable from u. Thus

$$\left|\Pr_x[M(a,x)=1 \mid E_v] - \Pr_x[M(a,x)=-1 \mid E_v]\right| = \left|(M \cdot \mathbb{P}_{x|v})(a)\right| \leq 2^{-2r},$$

and hence

$$\Pr_x[M(a,x) \neq b \mid E_v] \leq \tfrac{1}{2} + 2^{-2r}.$$

Hence, by the union bound,

$$c_e = \Pr_x[(x \notin \widetilde{\mathrm{Sig}}(v)) \wedge (M(a,x)=b) \mid E_v] \geq \tfrac{1}{2} - 2 \cdot 2^{-2r}.$$

If $i \in L_s$, then $(a,b) \notin \mathrm{VeryBad}(v)$, and we have $\Pr_x[M(a,x)=b \mid E_v] \geq 2^{-4\tilde{\ell}}$. Thus,

$$c_e = \Pr_x[(x \notin \widetilde{\mathrm{Sig}}(v)) \wedge (M(a,x)=b) \mid E_v] \geq 2^{-4\tilde{\ell}} - 2 \cdot 2^{-2\ell}\,. \qquad \blacktriangleleft$$

Bounding the Norm of $\mathbb{P}_{x|s}$

We will show that $\left\|\mathbb{P}_{x|s}\right\|_2$ cannot be too large. Towards this, we will first prove that for every edge e of B' that is traversed by $\mathcal{T}$ starting from s_{j-1} with probability larger than zero, $\left\|\mathbb{P}_{x|e}\right\|_2$ cannot be too large.

▷ Claim 15. For any edge e of B', such that $\Pr(e) > 0$,

$$\left\|\mathbb{P}_{x|e}\right\|_2 \leq 2^{5\tilde{\ell}} \cdot 2^{\ell_j} \cdot 2^{-n}.$$

Proof. Let $e = (v, u)$ be an edge of B', labeled by (a, b), and such that $\Pr(e) > 0$. Since $\Pr(e) > 0$, the vertex v is not significant (as otherwise $\mathcal{T}$ always stops on v and hence $\Pr(e) = 0$). Thus,

$$\left\|\mathbb{P}_{x|v}\right\|_2 \leq 2^{\ell_j} \cdot 2^{-n}.$$

By Claim 14, for any $x' \in X$,

$$\mathbb{P}_{x|e}(x') = \begin{cases} 0 & \text{if} \quad x' \in \widetilde{\mathrm{Sig}}(v) \quad \text{or} \quad M(a, x') \neq b \\ \mathbb{P}_{x|v}(x') \cdot c_e^{-1} & \text{if} \quad x' \notin \widetilde{\mathrm{Sig}}(v) \quad \text{and} \quad M(a, x') = b \end{cases}$$

where c_e satisfies $c_e \geq 2^{-5\tilde{\ell}}$. Thus,

$$\left\|\mathbb{P}_{x|e}\right\|_2 \leq c_e^{-1} \cdot \left\|\mathbb{P}_{x|v}\right\|_2 \leq 2^{5\tilde{\ell}} \cdot 2^{\ell_j} \cdot 2^{-n}$$ ◀

▷ Claim 16.

$$\left\|\mathbb{P}_{x|s}\right\|_2 \leq 2^{5\tilde{\ell}} \cdot 2^{\ell_j} \cdot 2^{-n}.$$

Proof. Let $\Gamma_{in}(s)$ be the set of all edges e of B', that are going into s, such that $\Pr(e) > 0$. Note that

$$\sum_{e \in \Gamma_{in}(s)} \Pr(e) = \Pr(s).$$

By the law of total probability, for every $x' \in X$,

$$\mathbb{P}_{x|s}(x') = \sum_{e \in \Gamma_{in}(s)} \frac{\Pr(e)}{\Pr(s)} \cdot \mathbb{P}_{x|e}(x'),$$

and hence by Jensen's inequality,

$$\mathbb{P}_{x|s}(x')^2 \leq \sum_{e \in \Gamma_{in}(s)} \frac{\Pr(e)}{\Pr(s)} \cdot \mathbb{P}_{x|e}(x')^2.$$

Summing over $x' \in X$, we obtain,

$$\left\|\mathbb{P}_{x|s}\right\|_2^2 \leq \sum_{e \in \Gamma_{in}(s)} \frac{\Pr(e)}{\Pr(s)} \cdot \left\|\mathbb{P}_{x|e}\right\|_2^2.$$

By Claim 15, for any $e \in \Gamma_{in}(s)$,

$$\left\|\mathbb{P}_{x|e}\right\|_2^2 \leq \left(2^{5\tilde{\ell}} \cdot 2^{\ell_j} \cdot 2^{-n}\right)^2.$$

Hence,

$$\left\|\mathbb{P}_{x|s}\right\|_2^2 \leq \left(2^{5\tilde{\ell}} \cdot 2^{\ell_j} \cdot 2^{-n}\right)^2.$$ ◀

Similarity to a Target Distribution

Recall that for two functions $f, g : X \to \mathbb{R}^+$, we defined

$$\langle f, g \rangle = \mathop{\mathbf{E}}_{z \in_R X}[f(z) \cdot g(z)].$$

We think of $\langle f, g \rangle$ as a measure for the similarity between a function f and a target function g. Typically f, g will be distributions.

▷ Claim 17.

$$\langle \mathbb{P}_{x|s}, \mathbb{P}_{x|s} \rangle > 2^{2\ell_j} \cdot 2^{-2n}.$$

Proof. Since s is significant,

$$\langle \mathbb{P}_{x|s}, \mathbb{P}_{x|s} \rangle = \left\| \mathbb{P}_{x|s} \right\|_2^2 > 2^{2\ell_j} \cdot 2^{-2n}.$$ ◀

▷ Claim 18.

$$\langle \mathcal{U}_X, \mathbb{P}_{x|s} \rangle = 2^{-2n},$$

where $\mathcal{U}_X$ is the uniform distribution over X.

Proof. Since $\mathbb{P}_{x|s}$ is a distribution,

$$\langle \mathcal{U}_X, \mathbb{P}_{x|s} \rangle = 2^{-2n} \cdot \sum_{z \in X} \mathbb{P}_{x|s}(z) = 2^{-2n}.$$ ◀

Measuring the Progress

For $i \in \{0, \ldots, m\}$, let

$$\mathcal{Z}_i = \sum_{v \in \mathcal{L}'_i} \Pr(v) \cdot \langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s} \rangle^k.$$

For $i \in \{1, \ldots, m\}$, let

$$\mathcal{Z}'_i = \sum_{e \in \Gamma_i} \Pr(e) \cdot \langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle^k.$$

We think of $\mathcal{Z}_i, \mathcal{Z}'_i$ as measuring the progress made by the branching program, towards reaching a state with distribution similar to $\mathbb{P}_{x|s}$.

For a vertex $v \in \mathcal{L}'_i$ of B', let $\Gamma_{out}(v)$ be the set of all edges e of B', that are going out of v to $\mathcal{L}'_{i+1}$, such that $\Pr(e) > 0$. Note that

$$\sum_{e \in \Gamma_{out}(v)} \Pr(e) \leq \Pr(v).$$

(We don't always have an equality here, since sometimes $\mathcal{T}$ stops on v, or goes to a vertex from which s is not reachable).

Recall that L_s stores a (not too long) list of indices to layers on which the path might choose to go over bad edges. The next four claims show that the progress made by the branching program is slow on every layer $i \notin L_s$. On layers $i \in L_s$ the progress might be significant but we will still have meaningful bounds on it.

▷ Claim 19. For every vertex $v \in \mathcal{L}'_{i-1}$, such that $\Pr(v) > 0$,

$$\sum_{e \in \Gamma_{out}(v)} \frac{\Pr(e)}{\Pr(v)} \cdot \langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle^k \leq \langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s} \rangle^k \cdot c_i^k + 2^{-2nk+k} \cdot c_i^k,$$

where c_i is defined as

- $c_i = 1 + 2^{-r}$, if $i \notin L_s$.
- $c_i = 2^{5\tilde{\ell}}$, if $i \in L_s$.

Proof. If v is significant or v is a leaf, then $\mathcal{T}$ always stops on v and hence $\Gamma_{out}(v)$ is empty and thus the left hand side is equal to zero and the right hand side is positive, so the claim follows trivially. Thus, we can assume that v is not significant and is not a leaf.

Define $P : X \to \mathbb{R}^+$ as follows. For any $x' \in X$,

$$P(x') = \begin{cases} 0 & \text{if} \quad x' \in \widetilde{\mathrm{Sig}}(v) \\ \mathbb{P}_{x|v}(x') & \text{if} \quad x' \notin \widetilde{\mathrm{Sig}}(v) \end{cases}$$

Note that by the definition of $\mathrm{Sig}(v)$ and since $\mathrm{Sig}(v) \subseteq \widetilde{\mathrm{Sig}}(v)$, for any $x' \in X$,

$$P(x') \leq 2^{4\ell} \cdot 2^{-n}. \tag{19}$$

Define $f : X \to \mathbb{R}^+$ as follows. For any $x' \in X$,

$$f(x') = P(x') \cdot \mathbb{P}_{x|s}(x').$$

By Claim 16 and Equation (19),

$$\|f\|_2 \leq 2^{4\ell} \cdot 2^{-n} \cdot \left\|\mathbb{P}_{x|s}\right\|_2 \leq 2^{4\ell} \cdot 2^{-n} \cdot 2^{5\tilde{\ell}} \cdot 2^{\ell_j} \cdot 2^{-n} \leq 2^{10\ell} \cdot 2^{-2n}. \tag{20}$$

By Claim 14, for any edge $e \in \Gamma_{out}(v)$, labeled by (a, b), for any $x' \in X$,

$$\mathbb{P}_{x|e}(x') = \begin{cases} 0 & \text{if} \quad M(a, x') \neq b \\ P(x') \cdot c_e^{-1} & \text{if} \quad M(a, x') = b \end{cases}$$

where c_e satisfies $c_e \geq \frac{1}{2} - 2 \cdot 2^{-2r}$ if $i \notin L_s$ and $c_e \geq 2^{-5\tilde{\ell}}$ if $i \in L_s$. Denote by c_v the minimal value that c_e can get for $e \in \Gamma_{out}(v)$. By the above, $c_v \geq 2^{-5\tilde{\ell}}$ and $c_v \geq \frac{1}{2} - 2 \cdot 2^{-2r}$ if $i \notin L_s$. Note that $c_v^{-1} \leq 2c_i$ in both cases (recall that $c_i = 2^{5\tilde{\ell}}$ for $i \in L_s$ and $c_i = 1 + 2^{-r}$ if $i \notin L_s$). Therefore, for any edge $e \in \Gamma_{out}(v)$, labeled by (a, b), for any $x' \in X$,

$$\mathbb{P}_{x|e}(x') \cdot \mathbb{P}_{x|s}(x') = \begin{cases} 0 & \text{if} \quad M(a, x') \neq b \\ f(x') \cdot c_e^{-1} & \text{if} \quad M(a, x') = b \end{cases}$$

and hence, we have

$$\begin{aligned} \langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle &= \mathop{\mathbf{E}}_{x' \in_R X}[\mathbb{P}_{x|e}(x') \cdot \mathbb{P}_{x|s}(x')] = \mathop{\mathbf{E}}_{x' \in_R X}[f(x') \cdot c_e^{-1} \cdot \mathbf{1}_{\{x' \in X \,:\, M(a,x')=b\}}] \\ &= \mathop{\mathbf{E}}_{x' \in_R X}\left[f(x') \cdot c_e^{-1} \cdot \tfrac{(1+b \cdot M(a,x'))}{2}\right] \leq (\|f\|_1 + b \cdot \langle M_a, f \rangle) \cdot (2c_v)^{-1}. \end{aligned} \tag{21}$$

We will now consider two cases:

Case I: $\|f\|_1 < 2^{-2n}$

In this case, we bound $|\langle M_a, f \rangle| \leq \|f\|_1$ (since f is non-negative and the entries of M are in $\{-1, 1\}$) and obtain for any edge $e \in \Gamma_{out}(v)$,

$$\langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s} \rangle < c_v^{-1} \cdot 2^{-2n} \leq 2c_i \cdot 2^{-2n}.$$

Since $\sum_{e \in \Gamma_{out}(v)} \frac{\Pr(e)}{\Pr(v)} \leq 1$, Claim 19 follows, as the left hand side of the claim is smaller than the second term on the right hand side.

Case II: $\|f\|_1 \geq 2^{-2n}$

For every $a \in A$, define

$$t(a) = \frac{|\langle M_a, f\rangle|}{\|f\|_1}.$$

By Equation (21),

$$\langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s}\rangle^k < \|f\|_1^k \cdot (1+t(a))^k \cdot (2c_v)^{-k} \tag{22}$$

Note that by the definitions of P and f,

$$\|f\|_1 = \mathop{\mathbf{E}}_{x' \in_R X}[f(x')] = \langle P, \mathbb{P}_{x|s}\rangle \leq \langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s}\rangle.$$

Note also that for every $a \in A$, there is at most one edge $e_{(a,1)} \in \Gamma_{out}(v)$, labeled by $(a,1)$, and at most one edge $e_{(a,-1)} \in \Gamma_{out}(v)$, labeled by $(a,-1)$, and we have

$$\frac{\Pr(e_{(a,1)})}{\Pr(v)} + \frac{\Pr(e_{(a,-1)})}{\Pr(v)} \leq \frac{1}{|A|},$$

since $\frac{1}{|A|}$ is the probability that the next sample read by the program is a. Thus, summing over all $e \in \Gamma_{out}(v)$, by Equation (22),

$$\sum_{e \in \Gamma_{out}(v)} \frac{\Pr(e)}{\Pr(v)} \cdot \langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s}\rangle^k < \langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s}\rangle^k \cdot \mathop{\mathbf{E}}_{a \in_R A}\left[(1+t(a))^k\right] \cdot (2c_v)^{-k}. \tag{23}$$

It remains to bound

$$\mathop{\mathbf{E}}_{a \in_R A}\left[(1+t(a))^k\right], \tag{24}$$

using the properties of the matrix M and the bounds on the L_2 versus L_1 norms of f.

By Equation (20) and the assumption that $\|f\|_1 \geq 2^{-2n}$ we get

$$\frac{\|f\|_2}{\|f\|_1} \leq 2^{10\ell}.$$

Since M is a $(10k, 10\ell)$-L_2-extractor with error 2^{-10r}, there are at most $2^{-10k} \cdot |A|$ rows $a \in A$ with $t(a) = \frac{|\langle M_a, f\rangle|}{\|f\|_1} \geq 2^{-10r}$. We bound the expectation in Equation (24), by splitting the expectation into two sums

$$\mathop{\mathbf{E}}_{a \in_R A}\left[(1+t(a))^k\right] = \frac{1}{|A|} \cdot \sum_{a\,:\,t(a) \leq 2^{-10r}} (1+t(a))^k + \frac{1}{|A|} \cdot \sum_{a\,:\,t(a) > 2^{-10r}} (1+t(a))^k. \tag{25}$$

We bound the first sum in Equation (25) by $(1+2^{-10r})^k$. As for the second sum in Equation (25), we know that it is a sum of at most $2^{-10k} \cdot |A|$ elements, and since for every $a \in A$, we have $t(a) \leq 1$, we have

$$\frac{1}{|A|} \cdot \sum_{a\,:\,t(a) > 2^{-10r}} (1+t(a))^k \leq 2^{-10k} \cdot 2^k \leq 2^{-2r}$$

(where in the last inequality we used the fact that $r \leq k$). Overall, we get

$$\mathop{\mathbf{E}}_{a \in_R A}\left[(1+t(a))^k\right] \leq (1+2^{-10r})^k + 2^{-2r} \leq (1+2^{-2r})^{k+1}. \tag{26}$$

Substituting Equation (26) into Equation (23), we obtain

$$\sum_{e\in\Gamma_{out}(v)} \frac{\Pr(e)}{\Pr(v)} \cdot \langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s}\rangle^k < \langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s}\rangle^k \cdot \left(1+2^{-2r}\right)^{k+1} \cdot (2c_v)^{-k}.$$

If $i \notin L_s$, then $(2c_v)^{-1} \leq \left(1+2^{-2r+3}\right)$ and thus $\left(1+2^{-2r}\right)^{k+1} \cdot (2c_v)^{-k} \leq (1+2^{-r})^k$ (where the inequality uses the assumption that r is sufficiently large).

If $i \in L_s$, then $(2c_v)^{-1} \leq \frac{1}{2} \cdot 2^{5\tilde{\ell}}$ and thus $\left(1+2^{-2r}\right)^{k+1} \cdot (2c_v)^{-k} \leq 2^{5\tilde{\ell}k}$. This completes the proof of Claim 19. ◁

▷ Claim 20. Recall the definition of c_i from Claim 19. For every $i \in \{1,\ldots,m\}$,

$$\mathcal{Z}'_i \leq (\mathcal{Z}_{i-1} + 2^{-2nk+k}) \cdot c_i^k$$

Proof. By Claim 19,

$$\begin{aligned}
\mathcal{Z}'_i = \sum_{e\in\Gamma_i} \Pr(e) \cdot \langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s}\rangle^k &= \sum_{v\in\mathcal{L}'_{i-1}} \Pr(v) \cdot \sum_{e\in\Gamma_{out}(v)} \frac{\Pr(e)}{\Pr(v)} \cdot \langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s}\rangle^k \\
&\leq \sum_{v\in\mathcal{L}'_{i-1}} \Pr(v) \cdot \left(\langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s}\rangle^k + 2^{-2nk+k}\right) \cdot c_i^k \\
&= c_i^k \cdot \left(\mathcal{Z}_{i-1} + \sum_{v\in\mathcal{L}'_{i-1}} \Pr(v) \cdot 2^{-2nk+k}\right) \\
&\leq c_i^k \cdot \left(\mathcal{Z}_{i-1} + 2^{-2nk+k}\right)
\end{aligned}$$

◀

▷ Claim 21. For every $i \in \{1,\ldots,m\}$,

$$\mathcal{Z}_i \leq \mathcal{Z}'_i.$$

Proof. For any $v \in \mathcal{L}'_i$, let $\Gamma_{in}(v)$ be the set of all edges $e \in \Gamma_i$, that are going into v. Note that

$$\sum_{e\in\Gamma_{in}(v)} \Pr(e) = \Pr(v).$$

By the law of total probability, for every $v \in \mathcal{L}'_i$ and every $x' \in X$,

$$\mathbb{P}_{x|v}(x') = \sum_{e\in\Gamma_{in}(v)} \frac{\Pr(e)}{\Pr(v)} \cdot \mathbb{P}_{x|e}(x'),$$

and hence

$$\langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s}\rangle = \sum_{e\in\Gamma_{in}(v)} \frac{\Pr(e)}{\Pr(v)} \cdot \langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s}\rangle.$$

Thus, by Jensen's inequality,

$$\langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s}\rangle^k \leq \sum_{e\in\Gamma_{in}(v)} \frac{\Pr(e)}{\Pr(v)} \cdot \langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s}\rangle^k.$$

Summing over all $v \in \mathcal{L}'_i$, we get

$$\begin{aligned}
\mathcal{Z}_i = \sum_{v\in\mathcal{L}'_i} \Pr(v) \cdot \langle \mathbb{P}_{x|v}, \mathbb{P}_{x|s}\rangle^k &\leq \sum_{v\in\mathcal{L}'_i} \Pr(v) \cdot \sum_{e\in\Gamma_{in}(v)} \frac{\Pr(e)}{\Pr(v)} \cdot \langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s}\rangle^k \\
&= \sum_{e\in\Gamma_i} \Pr(e) \cdot \langle \mathbb{P}_{x|e}, \mathbb{P}_{x|s}\rangle^k = \mathcal{Z}'_i.
\end{aligned}$$

◀

▷ Claim 22. For every $i \in \{1, \ldots, m\}$,

$$\mathcal{Z}_i \leq 2^{r+3k+5\tilde{\ell}k\cdot|L_s|} \cdot 2^{-2k\cdot n}.$$

Proof. By Claim 20 and Claim 21, for every $i \in \{1, \ldots, m\}$,

$$\mathcal{Z}_i \leq (\mathcal{Z}_{i-1} + 2^{-2nk+k}) \cdot c_i^k$$

where $c_i = (1 + 2^{-r})$ if $i \notin L_s$ and $c_i = 2^{5\tilde{\ell}}$ if $i \in L_s$. Thus, we can show by induction on $i \in \{1, \ldots, m\}$ that

$$\mathcal{Z}_i \leq 2^{-2nk+k} \cdot (i+1) \cdot \prod_{i'=1}^{i} c_{i'}^k$$

Hence, for any $i \in \{1, \ldots, m\}$ it holds that

$$\mathcal{Z}_i \leq 2^{-2nk+k} \cdot (m+1) \cdot (1 + 2^{-r})^{mk} \cdot 2^{5\tilde{\ell}k|L_s|}.$$

Since $m \leq 2^{\epsilon\tilde{r}} \leq 2^r - 1$,

$$\mathcal{Z}_i \leq 2^{-2k\cdot n+k} \cdot 2^r \cdot e^k \cdot 2^{5\tilde{\ell}k|L_s|}.$$ ◀

Proof of Lemma 7

We can now complete the proof of Lemma 7. Assume that s is in layer-i of B'. By Claim 17,

$$\mathcal{Z}_i \geq \Pr(s) \cdot \langle \mathbb{P}_{x|s}, \mathbb{P}_{x|s} \rangle^k > \Pr(s) \cdot \left(2^{2\ell_j} \cdot 2^{-2n}\right)^k = \Pr(s) \cdot 2^{2\ell_j \cdot k} \cdot 2^{-2k\cdot n}.$$

On the other hand, by Claim 22,

$$\mathcal{Z}_i \leq 2^{r+3k+5\tilde{\ell}k\cdot|L_s|} \cdot 2^{-2k\cdot n}.$$

Thus, we get

$$\Pr(s) \leq 2^{r+3k+5\tilde{\ell}k\cdot|L_s|} \cdot 2^{-2\ell_j\cdot k}$$

We treat differently the case $j = 1$ and $j = 2$ as follows. For $j = 1$, the set L_s is empty, and we have

$$\Pr(s) \leq 2^{r+3k} \cdot 2^{-2\ell_1\cdot k} \leq 2^{-\ell_1\cdot k}.$$

For $j = 2$ we have

$$\begin{aligned}
\Pr(s) &\leq 2^{r+3k+5\tilde{\ell}k\cdot|L_s|} \cdot 2^{-2\ell\cdot k} \\
&\leq 2^{r+3k+1000\epsilon\tilde{\ell}^2 k} \cdot 2^{-2\ell\cdot k} && (|L_s| \leq 200\epsilon\tilde{\ell}) \\
&\leq 2^{4k+1000\epsilon\ell k} \cdot 2^{-2\ell\cdot k} && (r \leq k,\ \tilde{\ell} \leq \sqrt{\ell}) \\
&\leq 2^{-\ell k} \leq 2^{-\ell_1\cdot k}\ . && (\epsilon < 1/10^{10})
\end{aligned}$$

Thus, in both cases we showed $\Pr(s) \leq 2^{-\ell_1 k}$.

Recall that we showed that the width of $\hat{B}$ is at most $2^{\epsilon k\tilde{\ell}/2}$, and note that the length of $\hat{B}$ is at most $2 \cdot 2^{\epsilon\tilde{r}}$. Taking a union bound over at most $2^{\epsilon k\tilde{\ell}/2} \cdot 2 \cdot 2^{\epsilon\tilde{r}} \leq 2^{k\ell_1/2}$ significant vertices of $\hat{B}$, we conclude that the probability that $\mathcal{T}$ reaches any significant vertex is at most $2^{-k\ell_1/2} = o(1)$. ◀

References

1 David A Barrington. Bounded-width polynomial-size branching programs recognize exactly those languages in NC1. *Journal of Computer and System Sciences*, 38(1):150–164, 1989.

2 Paul Beame, Shayan Oveis Gharan, and Xin Yang. Time-space tradeoffs for learning finite functions from random evaluations, with applications to polynomials. In *Conference On Learning Theory*, pages 843–856, 2018.

3 Benny Chor and Oded Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. *SIAM Journal on Computing*, 17(2):230–261, 1988.

4 Yuval Dagan and Ohad Shamir. Detecting Correlations with Little Memory and Communication. In *Conference On Learning Theory*, pages 1145–1198, 2018.

5 Sumegha Garg, Ran Raz, and Avishay Tal. Extractor-based time-space lower bounds for learning. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 990–1002. ACM, 2018.

6 Gillat Kol and Ran Raz. Interactive channel capacity. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 715–724. ACM, 2013.

7 Gillat Kol, Ran Raz, and Avishay Tal. Time-space hardness of learning sparse parities. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1067–1080. ACM, 2017.

8 Dana Moshkovitz and Michal Moshkovitz. Mixing implies lower bounds for space bounded learning. In *Conference on Learning Theory*, pages 1516–1566, 2017.

9 Dana Moshkovitz and Michal Moshkovitz. Entropy samplers and strong generic lower bounds for space bounded learning. In *9th Innovations in Theoretical Computer Science Conference (ITCS 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

10 Michal Moshkovitz and Naftali Tishby. Mixing complexity and its applications to neural networks. *arXiv preprint*, 2017. `arXiv:1703.00729`.

11 Ran Raz. Fast learning requires good memory: A time-space lower bound for parity learning. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*, pages 266–275. IEEE, 2016.

12 Ran Raz. A time-space lower bound for a large class of learning problems. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 732–742. IEEE, 2017.

13 Miklos Santha and Umesh V Vazirani. Generating quasi-random sequences from slightly-random sources. In *Foundations of Computer Science, 1984. 25th Annual Symposium on*, pages 434–440. IEEE, 1984.

14 Ohad Shamir. Fundamental limits of online and distributed algorithms for statistical learning and estimation. In *Advances in Neural Information Processing Systems*, pages 163–171, 2014.

15 Jacob Steinhardt, Gregory Valiant, and Stefan Wager. Memory, communication, and statistical queries. In *Conference on Learning Theory*, pages 1490–1516, 2016.

16 Gregory Valiant and Paul Valiant. Information theoretically secure databases. *arXiv preprint*, 2016. `arXiv:1605.02646`.

A Appendix

We first state the main theorem of [5] and then the modified proposition used in the proof of Lemma 9.

▶ **Theorem 23** (Theorem 1, [5]). *Let* $\frac{1}{100} < c < \frac{2}{3}$. *Fix* γ *to be such that* $\frac{3c}{2} < \gamma^2 < 1$. *Let* X, A *be two finite sets. Let* $n = \log_2 |X|$. *Let* $M : A \times X \to \{-1, 1\}$ *be a matrix which is a* (k', ℓ')*-*L_2*-extractor with error* $2^{-r'}$, *for sufficiently large*[1] k', ℓ' *and* r', *where* $\ell' \leq n$. *Let*

$$r := \min\left\{\frac{r'}{2}, \frac{(1-\gamma)k'}{2}, \frac{(1-\gamma)\ell'}{2} - 1\right\}.$$

[1] k', ℓ', r' are larger than some constant that depends on γ.

Let B be a branching program of length at most 2^r and width at most $2^{c\cdot k'\cdot \ell'}$ for the learning problem that corresponds to the matrix M. Then, the success probability of B is at most $O(2^{-r})$.

The authors prove the above theorem by first defining a truncated path that stops on a significant vertex, a significant value or a bad edge, such that, if the path doesn't stop before reaching a leaf, then the probability of guessing the correct x is small (at most $O(2^{-r})$ to be precise). Then, the authors prove that the probability that the truncated path stops is at most $O(2^{-r})$. Through slight modifications to the proof of the above theorem (with weaker bounds on the memory and length of B, in terms of constants), we can prove that the probability that a slightly modified truncated path stops is at most $2^{-\Omega(\min\{k',\ell'\})}$. As the modified proof is very similar to that of Theorem 23 and the original proof is lengthy, we just highlight the changes to the proof to get the following proposition.

▶ **Proposition 24.** *Let X, A be two finite sets. Let $n = \log_2|X|$. Let $M : A \times X \to \{-1,1\}$ be a matrix which is a (k',ℓ')-L_2-extractor with error $2^{-r'}$, for sufficiently large k',ℓ' and r', where $\ell' \le n$. Let*

$$r := \frac{\min\{r',k',\ell'\}}{100}.$$

Let B be a branching program of length at most 2^r and width at most $2^{\frac{k'\cdot\ell'}{100}}$ for the learning problem that corresponds to the matrix M. Then, there exists an event G such that

$$\Pr[G] \ge 1 - 2^{-\frac{\min\{k',\ell'\}}{8}}$$

and for every $x' \in X$ and every leaf z of the branching program B (with starting vertex z_0),

$$\Pr\left[x = x' \mid G \wedge (z_0 \widetilde{\to} z)\right] \le 2^{2\ell'}\cdot 2^{-n},$$

whenever the event $G \wedge (z_0 \widetilde{\to} z)$ is non-empty, where $z_0 \widetilde{\to} z$ denotes the event that the computational path (as opposed to the truncated path) from z_0 reaches z.

Proof. The proof of Theorem 1 of [5] defines the *truncated-path*, $\mathcal{T}$, to be the same as the computation-path of B, except that it sometimes stops before reaching a leaf. Roughly speaking, $\mathcal{T}$ stops before reaching a leaf if certain "bad" events occur. Nevertheless, the proof shows that the probability that $\mathcal{T}$ stops before reaching a leaf is negligible, so we can think of $\mathcal{T}$ as almost identical to the computation-path.

For a vertex v of B, we denote by E_v the event that $\mathcal{T}$ reaches the vertex v. We denote by $\Pr(v) = \Pr(E_v)$ the probability for E_v, and we denote by $\mathbb{P}_{x|v} = \mathbb{P}_{x|E_v}$ the distribution of the random variable x conditioned on the event E_v.

We first look at the definition of the truncated-path from the proof of Theorem 1 of [5]. We modify the stopping rules for a path as follows:

Let $\hat{l} = \frac{\ell'}{6}$.

Significant Vertices. A vertex v in layer-i of B is significant if

$$\left\|\mathbb{P}_{x|v}\right\|_2 > 2^{\hat{l}}\cdot 2^{-n}.$$

Significant Values. Even if v is not significant, $\mathbb{P}_{x|v}$ may have relatively large values. For a vertex v in layer-i of B, denote by $\mathrm{Sig}(v)$ the set of all $x' \in X$, such that,

$$\mathbb{P}_{x|v}(x') > 2^{3\hat{l}}\cdot 2^{-n}.$$

Bad Edges. For a vertex v in layer-i of B, denote by $\mathrm{Bad}(v)$ the set of all $\alpha \in A$, such that,

$$\left|(M \cdot \mathbb{P}_{x|v})(\alpha)\right| \geq 2^{-r'}.$$

Recall, that the truncated path is defined by induction on the layers of the branching program B:

The Truncated-Path $\mathcal{T}$

Assume that we already defined $\mathcal{T}$ until it reaches a vertex v in layer-i of B. The path $\mathcal{T}$ stops on v if (at least) one of the following occurs:

1. v is significant.
2. $x \in \mathrm{Sig}(v)$.
3. $a_{i+1} \in \mathrm{Bad}(v)$.
4. v is a leaf.

Otherwise, $\mathcal{T}$ proceeds by following the edge labeled by (a_{i+1}, b_{i+1}) (same as the computational-path).

The Event G

We define G to be the event that the truncated-path $\mathcal{T}$ didn't stop because of one of the first three stopping rules: That is, $\mathcal{T}$ didn't stop before reaching a leaf and didn't violate the significant vertices and significant values stopping rules (that is, the first two stopping rules) on the leaf that it reached.

We can upper bound the probability for $\bar{G}$ similarly to the way that it's done in [5].

▶ **Lemma 25.** *The probability that $\mathcal{T}$ reaches a significant vertex is at most $2^{-k'}$.*

The proof of the above lemma is very similar to the analogous lemma in the proof of Theorem 23. The only change is in the definition of significant value - we define the significant values to be the set of all $x' \in X$, such that, $\mathbb{P}_{x|v}(x') > 2^{3\hat{l}} \cdot 2^{-n}$ instead of the set of all $x' \in X$, such that, $\mathbb{P}_{x|v}(x') > 2^{2\hat{l}+2r} \cdot 2^{-n}$. With the above (worse in terms of constants) bounds on the memory and the length of the branching program, the proof works in the same way.

Lemma 25 shows that the probability that $\mathcal{T}$ stops on a vertex, because of the first reason (i.e., that the vertex is significant), is small. The next two claims imply that the probabilities that $\mathcal{T}$ stops on a vertex, because of the second and third reasons, are also small.

▷ Claim 26. If v is a non-significant vertex of B then

$$\Pr_x[x \in \mathrm{Sig}(v) \mid E_v] \leq 2^{-\hat{l}}.$$

Proof. Since v is not significant,

$$\mathop{\mathbf{E}}_{x' \sim \mathbb{P}_{x|v}}\left[\mathbb{P}_{x|v}(x')\right] = \sum_{x' \in X}\left[\mathbb{P}_{x|v}(x')^2\right] = 2^n \cdot \mathop{\mathbf{E}}_{x' \in_R X}\left[\mathbb{P}_{x|v}(x')^2\right] \leq 2^{2\hat{l}} \cdot 2^{-n}.$$

Hence, by Markov's inequality,

$$\Pr_{x' \sim \mathbb{P}_{x|v}}\left[\mathbb{P}_{x|v}(x') > 2^{\hat{l}} \cdot 2^{2\hat{l}} \cdot 2^{-n}\right] \leq 2^{-\hat{l}}.$$

Since conditioned on E_v, the distribution of x is $\mathbb{P}_{x|v}$, we obtain

$$\Pr_x\left[x \in \mathrm{Sig}(v) \mid E_v\right] = \Pr_x\left[\left(\mathbb{P}_{x|v}(x) > 2^{\hat{l}} \cdot 2^{2\hat{l}} \cdot 2^{-n}\right) \mid E_v\right] \leq 2^{-\hat{l}}. \quad ◀$$

▷ Claim 27. If v is a non-significant vertex of B then

$$\Pr_{a_{i+1}}[a_{i+1} \in \mathrm{Bad}(v)] \leq 2^{-k'}.$$

Proof. Since v is not significant, $\left\|\mathbb{P}_{x|v}\right\|_2 \leq 2^{\hat{I}} \cdot 2^{-n}$. Since $\mathbb{P}_{x|v}$ is a distribution, $\left\|\mathbb{P}_{x|v}\right\|_1 = 2^{-n}$. Thus,

$$\frac{\left\|\mathbb{P}_{x|v}\right\|_2}{\left\|\mathbb{P}_{x|v}\right\|_1} \leq 2^{\hat{I}} \leq 2^{\ell'}.$$

Since M is a (k', ℓ')-L_2-extractor with error $2^{-r'}$, there are at most $2^{-k'} \cdot |A|$ elements $\alpha \in A$ with

$$\left|\langle M_\alpha, \mathbb{P}_{x|v}\rangle\right| \geq 2^{-r'} \cdot \left\|\mathbb{P}_{x|v}\right\|_1 = 2^{-r'} \cdot 2^{-n}.$$

The claim follows since a_{i+1} is uniformly distributed over A. ◁

We can now use Lemma 25, Claim 26 and Claim 27 to prove that the probability that $\mathcal{T}$ stops because of the first three stopping rules is at most $2^{-\frac{\min\{k',\ell'\}}{8}}$. Lemma 25 shows that the probability that $\mathcal{T}$ reaches a significant vertex and hence stops because of the first stopping rule, is at most $2^{-k'}$. Assuming that $\mathcal{T}$ doesn't reach any significant vertex (in which case it would have stopped because of the first stopping rule), Claim 26 shows that in each step, the probability that $\mathcal{T}$ stops because of the second stopping rule, is at most $2^{-\frac{\ell'}{6}}$. Taking a union bound over the 2^r steps, the total probability that $\mathcal{T}$ stops because of the second stopping rule, is at most $2^{-\frac{\ell'}{7}}$ (for sufficiently large ℓ'). In the same way, assuming that $\mathcal{T}$ doesn't reach any significant vertex (in which case it would have stopped because of the first stopping rule), Claim 27 shows that in each step, the probability that $\mathcal{T}$ stops because of the third stopping rule, is at most $2^{-k'}$. Again, taking a union bound over the 2^r steps, the total probability that $\mathcal{T}$ stops because of the third stopping rule, is at most $2^{-\frac{k'}{7}}$. Thus, the total probability that $\mathcal{T}$ stops (for any reason) before reaching a leaf (or violated the significant vertices or significant values stopping rules (that is, the first two stopping rules) on the leaf that it reached) is at most $2^{-\frac{\min\{k',\ell'\}}{8}}$. (Summing over the three probabilities and using the fact that k', ℓ' are sufficiently large).

Thus, $\Pr[\bar{G}] \leq 2^{-\frac{\min\{k',\ell'\}}{8}}$, as required.

Bounding $\Pr[x = x' \mid G \wedge (z_0 \widetilde{\rightarrow} z)]$

It remains to prove that for every $x' \in X$ and every leaf z of the branching program B (with starting vertex z_0),

$$\Pr[x = x' \mid G \wedge (z_0 \widetilde{\rightarrow} z)] \leq 2^{2\ell'} \cdot 2^{-n},$$

whenever the event $G \wedge (z_0 \widetilde{\rightarrow} z)$ is non-empty, where $z_0 \widetilde{\rightarrow} z$ denotes the event that the computational path (as opposed to the truncated path) from z_0 reaches z.

Recall that E_z is the event that $\mathcal{T}$ reaches the vertex z.

▷ Claim 28. The event $G \wedge (z_0 \widetilde{\rightarrow} z)$ is equivalent to $E_z \wedge (z \text{ is not significant}) \wedge (x \notin \mathrm{Sig}(z))$.

Proof. If $G \wedge (z_0 \widetilde{\rightarrow} z)$ occurs then the truncated-path $\mathcal{T}$ didn't stop before reaching a leaf (since G occurs) and the computational path from z_0 reaches z (since $(z_0 \widetilde{\rightarrow} z)$ occurs). Thus, E_z occurs. Also, since the first stopping rule is not violated on z, we have that z is not significant and since the second stopping rule is not violated on z, we have $x \notin \mathrm{Sig}(z)$.

On the other direction, if $E_z \wedge (z \text{ is not significant}) \wedge (x \notin \mathrm{Sig}(z))$ occurs, then $(z_0 \widetilde{\rightarrow} z)$ occurs (since E_z occurs), the truncated-path $\mathcal{T}$ didn't stop before reaching a leaf (since E_z occurs) and none of the first two stopping rules are violated on z, since z is not significant and $x \notin \mathrm{Sig}(z)$. ◁

By Claim 28, it remains to prove that for every leaf z and every $x' \in X$, if the event $E_z \wedge (z \text{ is not significant}) \wedge (x \notin \mathrm{Sig}(z))$ is non-empty then

$$\Pr\left[x = x' \mid E_z \wedge (z \text{ is not significant}) \wedge (x \notin \mathrm{Sig}(z))\right] \leq 2^{2\ell'} \cdot 2^{-n}.$$

Equivalently, we need to prove that for every non-significant leaf z and every $x' \in X$, if the event $E_z \wedge (x \notin \mathrm{Sig}(z))$ is non-empty then

$$\mathbb{P}_{x|E_z \wedge (x \notin \mathrm{Sig}(z))}(x') = \Pr\left[x = x' \mid E_z \wedge (x \notin \mathrm{Sig}(z))\right] \leq 2^{2\ell'} \cdot 2^{-n}.$$

By the definition of conditional distribution,

$$\mathbb{P}_{x|E_z \wedge (x \notin \mathrm{Sig}(z))}(x') = \begin{cases} 0 & \text{if} \quad x' \in \mathrm{Sig}(z) \\ \mathbb{P}_{x|E_z}(x') \cdot c^{-1} & \text{if} \quad x' \notin \mathrm{Sig}(z) \end{cases}$$

where $c = \sum_{x' \notin \mathrm{Sig}(z)} \mathbb{P}_{x|E_z}(x')$ is the normalization factor. As z is not significant, by Claim 26,

$$\Pr_x[x \in \mathrm{Sig}(z) \mid E_z] \leq 2^{-\hat{l}}.$$

Therefore, $c \geq 1 - 2^{-\hat{l}}$. Since by the definition of $\mathrm{Sig}(z)$, for $x' \notin \mathrm{Sig}(z)$, we have $\mathbb{P}_{x|z}(x') \leq 2^{3\hat{l}} \cdot 2^{-n}$, we can bound

$$\mathbb{P}_{x|E_z \wedge (x \notin \mathrm{Sig}(z))}(x') \leq 2^{3\hat{l}} \cdot 2^{-n} \cdot c^{-1} \leq 2^{3\hat{l}+1} \cdot 2^{-n} \leq 2^{2\ell'} \cdot 2^{-n}.$$ ◀

Parity Helps to Compute Majority

Igor Carboni Oliveira
Department of Computer Science, University of Oxford, UK
igor.carboni.oliveira@cs.ox.ac.uk

Rahul Santhanam
Department of Computer Science, University of Oxford, UK
rahul.santhanam@cs.ox.ac.uk

Srikanth Srinivasan
Department of Mathematics, IIT Bombay, India
srikanth@math.iitb.ac.in

Abstract

We study the complexity of computing symmetric and threshold functions by constant-depth circuits with Parity gates, also known as $\mathsf{AC}^0[\oplus]$ circuits. Razborov [23] and Smolensky [25, 26] showed that Majority requires depth-d $\mathsf{AC}^0[\oplus]$ circuits of size $2^{\Omega(n^{1/2(d-1)})}$. By using a divide-and-conquer approach, it is easy to show that Majority can be computed with depth-d $\mathsf{AC}^0[\oplus]$ circuits of size $2^{\widetilde{O}(n^{1/(d-1)})}$. This gap between upper and lower bounds has stood for nearly three decades.

Somewhat surprisingly, we show that *neither* the upper bound nor the lower bound above is tight for large d. We show for $d \geq 5$ that any symmetric function can be computed with depth-d $\mathsf{AC}^0[\oplus]$ circuits of size $\exp(\widetilde{O}(n^{\frac{2}{3}\cdot\frac{1}{(d-4)}}))$. Our upper bound extends to threshold functions (with a constant additive loss in the denominator of the double exponent). We improve the Razborov-Smolensky lower bound to show that for $d \geq 3$ Majority requires depth-d $\mathsf{AC}^0[\oplus]$ circuits of size $2^{\Omega(n^{1/(2d-4)})}$. For depths $d \leq 4$, we are able to refine our techniques to get almost-optimal bounds: the depth-3 $\mathsf{AC}^0[\oplus]$ circuit size of Majority is $2^{\widetilde{\Theta}(n^{1/2})}$, while its depth-4 $\mathsf{AC}^0[\oplus]$ circuit size is $2^{\widetilde{\Theta}(n^{1/4})}$.

2012 ACM Subject Classification Theory of computation

Keywords and phrases Computational Complexity, Boolean Circuits, Lower Bounds, Parity, Majority

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.23

Funding This work was supported in part by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2014)/ERC Grant Agreement no. 615075.

Acknowledgements This paper is the result of a collaboration that happened at the Simons Institute program on Lower Bounds in Computational Complexity, where all three authors were long-term visitors. We are grateful to the Simons Institute for their support.

1 Introduction

Given the difficulty of proving lower bounds for general Boolean circuits, much work in circuit complexity has focused on restricted classes, and in particular on bounded-depth classes. Super-polynomial lower bounds are known for explicit Boolean functions against various classes of bounded-depth circuit classes, including AC^0 (constant-depth circuits with unbounded fan-in AND and OR gates) and $\mathsf{AC}^0[\oplus]$ (constant-depth circuits with unbounded fan-in AND, OR and Parity gates).

In the case of AC^0, we have almost optimal size bounds [1, 11, 14] for the Parity function. A simple divide-and-conquer argument shows that Parity on n variables can be computed by depth-d AC^0 circuits of size $\widetilde{O}(2^{n^{1/(d-1)}})$. The classic lower bound of Håstad [14] shows that Parity requires depth-d AC^0 circuits of size $2^{\Omega(n^{1/(d-1)})}$. Thus the upper bound is tight up to the constant factor in the exponent. The same lower bound holds for the Majority

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka; Article No. 23; pp. 23:1–23:17


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

function [14], but the upper bound given by the divide-and-conquer argument [8] weakens slightly to $2^{\widetilde{O}(n^{1/(d-1)})}$, meaning that the upper bound is tight up to a logarithmic factor in the exponent.

For $\mathsf{AC}^0[\oplus]$, however, we do not have optimal bounds. The celebrated polynomial approximation method of Razborov and Smolensky [23, 25, 26] yields a lower bound of $2^{n^{1/2(d-1)}}$ on the size of depth-d $\mathsf{AC}^0[\oplus]$ circuits computing Majority. The best known upper bound thus far for Majority was the one mentioned in the previous paragraph, which in fact gives constant-depth circuits that don't use Parity gates.

Note that there is a significant gap between upper and lower bounds for Majority – the exponent in the upper bound is quadratically larger than the exponent in the lower bound. This gap between upper and lower bounds has stood for almost three decades.

Since the best known upper bound for Majority can be implemented without Parity gates, a natural question arises. Do Parity gates help when computing Majority using bounded-depth circuits? It is easy to see that Majority gates help to compute Parity – indeed, Parity can be easily written as a small DNF of Majorities. However, it is far from clear how to take advantage of Parity to compute Majority. Indeed, we ourselves believed until recently that the upper bound was close to optimal for $\mathsf{AC}^0[\oplus]$ circuits computing Majority.

1.1 Our results

Our main result in this paper is that *neither* the upper bound using divide-and-conquer nor the lower bound given by the Razborov-Smolensky method is tight for Majority, when the depth is large enough. We first describe our new upper bound, and then our lower bound that slightly improves Razborov-Smolensky.

First, we show how to save a constant factor in the double exponent when computing Majority.

▶ **Theorem 1.** *Let $d \geq 5$ be an integer. Majority on n bits can be computed by depth-d $\mathsf{AC}^0[\oplus]$ circuits of size $2^{\widetilde{O}\left(n^{\frac{2}{3}\cdot\frac{1}{(d-4)}}\right)}$.*

Theorem 1 follows from a result giving the same upper bound for the $\mathsf{AC}^0[\oplus]$ size complexity of any *symmetric* function. Similar techniques combined with another idea and a more careful implementation allow us to obtain an improved upper bound at depth $d = 4$ (stated in Corollary 3 below). We also show how to extend the upper bound in Theorem 1 to any *linear threshold function*, though we lose a small additive term in the denominator of the double exponent. We refer to the body of the paper for more details about the latter result.

Next, we show how to improve the Razborov-Smolensky lower bound slightly to achieve a better double exponent.

▶ **Theorem 2.** *For any integer $d \geq 3$, Majority on n bits requires depth-d $\mathsf{AC}^0[\oplus]$ circuits of size $2^{\Omega(n^{1/(2d-4)})}$.*

Note that there is still a gap between our new upper bound for Majority given by Theorem 1 and our new lower bound given by Theorem 2. We do not have a clear belief at this point about what the optimal size bound should be at large depths.

For depths $d = 3$ and $d = 4$, our results do provide nearly optimal bounds. For $d = 3$, the new lower bound is close to the upper bound given by the divide-and-conquer strategy, showing that parity gates do no significantly help when $d = 3$. On the other hand, for depth $d = 4$ our improved upper bound construction essentially matches the new lower bound from Theorem 2.

▶ **Corollary 3.** *The following results hold*:

(i) *The depth-3* $\mathsf{AC}^0[\oplus]$ *circuit size complexity of Majority is* $2^{\widetilde{\Theta}(n^{1/2})}$.

(ii) *The depth-4* $\mathsf{AC}^0[\oplus]$ *circuit size complexity of Majority is* $2^{\widetilde{\Theta}(n^{1/4})}$.

Note the contrast with the known size bounds for AC^0 circuits, as a result of which we have a significantly stronger $\mathsf{AC}^0[\oplus]$ upper bound than an AC^0 lower bound at depth $d = 4$, but not at smaller depths.

Our results indicate that even for simple circuit models, naive upper bound strategies might not be optimal, and surprising savings can be achieved in circuit size. It might be worthwhile to look for other examples of this phenomenon.

1.2 Proof ideas

Upper bounds. We describe the upper bound for the Majority function (the same idea works for any symmetric function.) We follow the same basic high-level strategy as a construction of better approximating polynomials for the Majority function due to Alman and Williams [3]. They observed that while the Majority function on n variables seems hardest to compute when the Hamming weight is close to $n/2$, by polynomial interpolation, it is easy to obtain a low-degree polynomial that computes Majority on such inputs. Conversely, when the input has weight far from $n/2$, one can use sampling to reduce the input size and recurse.

A similar idea works in our setting as well. For inputs of weight within distance t of $n/2$, we use a degree-t polynomial to compute the Majority function. This polynomial is over $\mathbb{F}_2$ and moreover *symmetric*, and thus by standard techniques can be represented as a $\mathsf{AC}^0[\oplus]$ circuit of depth d and size roughly $\exp(t^{2/d} \log n)$. When the weight is t-far from $n/2$, we use sampling not to recurse but to solve the problem directly. In fact, using standard results on the complexity of the *Coin Problem* from the literature [21, 4], it follows that there are AC^0 circuits that solve Majority on inputs that are t-far from $n/2$ in depth d and size roughly $\exp((n/t)^{1/d})$. Putting these strategies together and optimizing the value of t yields the upper bound.

The above strategy yields a constant factor improvement in the double exponent of known upper bounds for large enough d. Using these ideas and a bit more work, we are also able to obtain a similar improvement at depth 4. In particular, all these upper bounds are stronger than known AC^0 *lower bounds* for symmetric functions [14], proving that parity gates indeed help in computing arbitrary symmetric functions.

It is worth understanding what these upper bounds mean at a higher level. A possible comparison can be made with the well-known result of Barrington, Beigel and Rudich [6], which showed that the OR function on n variables can be represented by a polynomial modulo 6 of degree just $O(\sqrt{n})$. The crucial observation there was that given any two distinct integers $i, j \in \{0, \ldots, n\}$, their difference $i - j$ cannot simultaneously be divisible by a large power of 2 and a large power of 3 (here, "large" means more than $\sqrt{n}$). This can be stated in the language of *p-adic norms*: recall that for a prime p, the p-adic distance between i and j is inversely related to the largest power of p that divides $i - j$. Thus, the result of [6] uses the fact that no i, j as above can be at small 2-adic as well as 3-adic distances. Our result leverages a similar contrast between the 2-adic norm and the standard Euclidean norm.

Lower Bounds. First we describe a new but weaker circuit size lower bound of $2^{\Omega(n^{1/(2d-3)})}$. Our proof follows the general polynomial approximation framework of Razborov [23]. The high-level idea is to show that any $\mathsf{AC}^0[\oplus]$ circuit of small size can be approximated by low-degree polynomials from $\mathbb{F}_2[x_1, \ldots, x_n]$: this is done by approximating each AND/OR

gate[1] in the circuit by a low-degree polynomial and composing these approximations together. The second step is to show that the hard function (the Majority function in our scenario) does not have low-degree polynomial approximations of this form; here, the proofs that yield the best known parameters are due to Smolensky [25, 26].

To improve on known lower bounds, we need two new ingredients.

1. The first is the observation that the standard Razborov approximations for the OR and AND functions are *one-sided*. While this is obvious from the construction, we do not know of a previous lower-bound application of this. In our setting, we use this to show that any $\mathsf{AC}^0[\oplus]$ circuit C has a low-degree polynomial approximation P where the approximation is much better on one of $C^{-1}(0)$ or $C^{-1}(1)$.
2. To use these improved polynomial approximations, we need an improved lower bound for approximating the Majority function in the sense described above. It follows from Smolensky's work [25] that any polynomial that computes the Majority function on all but an ε-fraction of inputs must have degree $\Omega(\sqrt{n\log(1/\varepsilon)})$. In our setting, however, we need to lower bound the degree of a polynomial computing the Majority function on all but an ε-fraction of the 0-inputs but may err on a constant (say $1/10$) fraction of the 1-inputs. We are able to recover the lower bound of $\Omega(\sqrt{n\log(1/\varepsilon)})$ even under these weaker assumptions, which finishes the proof. This extension of Smolensky's lower bound uses results on the combinatorics of Hilbert functions [29, 15, 20].

Using some of the above ideas in conjunction with standard AC^0 lower bound techniques [1, 11, 14] based on random restrictions, we show how to get a lower bound of $\exp(\Omega(\sqrt{n}))$ on the size of depth-3 circuits for the Majority function, matching the known AC^0 lower bound [14] and nearly matching the AC^0 upper bound of $\exp(\tilde{O}(\sqrt{n}))$ [8].

Finally, we observe that the method of random restrictions employed in the depth-3 lower bound allows us to further improve the lower bound in the general case. To achieve that, we combine the refined analysis of approximate degree from items 1. and 2. above with the effects of a random restriction on the approximate degree of depth-2 subcircuits. This gives a $2^{\Omega(n^{1/(2d-4)})}$ lower bound on the depth-d $\mathsf{AC}^0[\oplus]$ circuit size of Majority, completing the proof of Theorem 2.

2 The Upper Bounds

2.1 An improved upper bound for all large depths

▶ **Theorem 4.** *For every integer $d \geq 5$, if $f_n\colon \{0,1\}^n \to \{0,1\}$ is a symmetric boolean function then it can be computed by an $\mathsf{AC}^0[\oplus]$ circuit of depth d and of size $2^{\widetilde{O}\left(n^{\frac{2}{3}\cdot\frac{1}{(d-4)}}\right)}$.*

Proof. For convenience, given a string $x \in \{0,1\}^*$, we let $|x|_1 \stackrel{\text{def}}{=} \sum_i x_i$ denote its hamming weight. For $0 \leq i, j \leq n$ and $i \neq j$, let $D_{i,j}$ and E_i be boolean functions on n-bit inputs satisfying

$$D_{i,j}(y) = \begin{cases} 1 & \text{if } |y|_1 = i, \\ 0 & \text{if } |y|_1 = j. \end{cases} \qquad E_i(y) = \begin{cases} 1 & \text{if } |y|_1 = i, \\ 0 & \text{otherwise.} \end{cases}$$

[1] The parity gates are already low-degree polynomials and hence trivially approximable in this sense.

(The behaviour of $D_{i,j}$ on inputs of different hamming weight is not relevant in our construction.) Notice that, for every $0 \leq i \leq n$,

$$E_i(y) = \bigwedge_{\substack{0 \leq j \leq n \\ j \neq i}} D_{i,j}(y).$$

Clearly, if $f_n \colon \{0,1\}^n \to \{0,1\}$ is a symmetric boolean function, then it can be written as a disjunction of at most $n+1$ functions E_i. (In other words, separating two layers of the hypercube can be as difficult as computing the hardest symmetric function.) Consequently, our task is reduced to the construction of $\mathsf{AC}^0[\oplus]$ circuits for each (partial) boolean function $D_{i,j}$.

Given $0 \leq i, j \leq n$ with $i \neq j$, we describe two circuits that agree with $D_{i,j}$ over the relevant input strings. The first circuit relies on an "algebraic" construction, while the second circuit is of a more "combinatorial" nature. Before going into further details, let us informally describe the main properties of the circuits (the discussion omits polylogarithmic factors in exponents).

1. Algebraic construction. If $|i-j| \leq n^{1/4}$, there is an $\mathsf{AC}^0[\oplus]$ circuit for $D_{i,j}$ of depth d and size roughly $2^{n^{1/2d}}$. This circuit explores parity gates in a crucial way.
2. Combinatorial construction. If $|i-j| \geq n^{1/2}$, there is an AC^0 circuit for $D_{i,j}$ of depth d and size roughly $2^{n^{1/2d}}$. This circuit is obtained from AND/OR circuits solving the coin problem.
3. In the "critical" interval $n^{1/4} \leq |i-j| \leq n^{1/2}$, we still don't know if there are circuits computing $D_{i,j}$ of size roughly $2^{n^{1/2d}}$. Jumping ahead, we will rely on the algebraic construction when $n^{1/4} \leq |i-j| \leq n^{1/3}$, and on the combinatorial construction when $n^{1/3} \leq |i-j| \leq n^{1/2}$. The maximum circuit complexity peaks at $|i-j| = n^{1/3}$, where both constructions provide depth-d circuits of size roughly $2^{n^{2/3d}}$.

We now present the technical details.

$\mathsf{AC}^0[\oplus]$ circuits for small $|i-j|$. We need the following lemma.

▶ **Lemma 5** ([3, Proof of Lemma 3.1]). *For integers $n \geq 1$, $k \geq 0$, and $\ell \geq 1$ such that $n \geq k+\ell-1$, and for every $c_0, \ldots, c_{\ell-1} \in \mathbb{Z}$, there is a multivariate polynomial $Q \colon \{0,1\}^n \to \mathbb{Z}$ of degree at most $\ell-1$ and with integer coefficients such that $p(x) = c_t$ for every $x \in \{0,1\}^n$ for which $|x|_1 = k+t$, where $0 \leq t \leq \ell-1$. Moreover,*

$$Q(x) = \sum_{t=0}^{\ell-1} a_t \cdot Q_t(x),$$

where each $a_t \in \mathbb{Z}$, and $Q_t(x) = \sum_{S \in \binom{[n]}{t}} \prod_{j \in S} x_j$ denotes the t-th elementary symmetric polynomial.

Since the function $\psi \colon \mathbb{Z} \to \mathbb{F}_2$ that maps each integer to its parity is a ring homomorphism, it follows from Lemma 5 that there is a polynomial $P \in \mathbb{F}_2[y_1, \ldots, y_n]$ of degree $\ell \leq |i-j|$ that agrees with $D_{i,j}$ on every input $y \in \{0,1\}^n$ such that $|y|_1 \in \{i, j\}$. Moreover,

$$P(y) = \sum_{t=0}^{\ell} b_t \cdot P_t(y),$$

where each $b_t \in \{0,1\}$, $P_t \in \mathbb{F}_2[y_1, \ldots, y_n]$ is the t-th elementary symmetric polynomial (over $\mathbb{F}_2$), and $t \leq \ell$.

It is well known that each polynomial P_t can be computed by an *algebraic branching program* of width n and length $t+1$ (the j-th layer contains nodes $1,\ldots,n$ that store the largest coordinate in $[n]$ that has been read so far). Using a standard divide-and-conquer approach which is analogous to the construction of bounded-depth circuits for distance-k connectivity (see e.g. [10]), it follows that for every even depth $d' \geq 2$, P_t can be computed by a layered $\mathsf{AC}^0[\oplus]$ circuit of depth d' (consisting of $\oplus$ and $\wedge$ gates) and of size at most $n^{O(t^{2/d'})}$. Moreover, the top gate of this circuit is a parity gate. Using the definition of P as a sum of polynomials P_t over $\mathbb{F}_2$ (which allows us to collapse two layers of parities), and the fact that $t \leq \ell \leq |i-j|$, it follows that for every even integer $d' \geq 2$, each $D_{i,j}$ can be computed by a depth-d' $\mathsf{AC}^0[\oplus]$ circuit of size at most $n^{O(|i-j|^{2/d'})}$. Consequently, there are circuits for $D_{i,j}$ of size $n^{O(|i-j|^{2/(d'-1)})}$ and depth d' for each integer $d' \geq 3$.

AC^0 circuits for large $|i-j|$. We assume without loss of generality that $i > j$, since negating the output of the circuit will handle the other case. First, we note that the computation of $D_{i,j}$ can be reduced to the case where i and j are near the middle layer. For $0 \leq a < b \leq m$, let $\mathsf{Promise\text{-}Th}^m_{a,b}$ be an m-bit boolean function such that

$$\mathsf{Promise\text{-}Th}^m_{a,b}(y) = \begin{cases} 0 & \text{if } |y|_1 \leq a, \\ 1 & \text{if } |y|_1 \geq b. \end{cases}$$

Let $r \stackrel{\text{def}}{=} i-j$. Then $D_{i,j}$ can be obtained as a projection of $\mathsf{Promise\text{-}Th}^{10n}_{5n-\lceil r/10\rceil,5n+\lceil r/10\rceil}$. More precisely, it is easy to check that, over the inputs of interest for $D_{i,j}$,

$$D_{i,j}(y) \;=\; \mathsf{Promise\text{-}Th}^{10n}_{5n-\lceil r/10\rceil,5n+\lceil r/10\rceil}(1^{5n-\lceil r/10\rceil-j}\, y\, 0^{10n-(5n-\lceil r/10\rceil-j+n)}).$$

It follows from the work of [24] (see also [18]) that this promise threshold function can be computed by *randomized* AC^0 circuits of depth $d' \geq 2$ and of size $\exp(O(1/\delta)^{1/(d'-1)})$, where $\delta \stackrel{\text{def}}{=} \Theta(r/n)$. By a standard derandomization argument (see e.g. [2]) that increases the number of layers by at most 2, and by collapsing adjacent layers during this derandomization, it follows that for every $d' \geq 3$, $\mathsf{Promise\text{-}Th}^{10n}_{5n-\lceil r/10\rceil,5n+\lceil r/10\rceil}$ can be computed by a (deterministic) depth-d' circuit of size $\exp(O(n/r)^{1/(d'-2)})$. Therefore, for every integer $d' \geq 3$, each $D_{i,j}$ can be computed by a depth-d' AC^0 circuit of size at most $\exp(O(n/|i-j|)^{1/(d'-2)})$.

Let $d \geq 5$ be given. In order to compute the symmetric function f_n, we proceed as described above. Two layers are employed to combine the sub-circuits $D_{i,j}$ in the appropriate way (via the functions E_i). In the remaining $d' \stackrel{\text{def}}{=} d-2$ layers, we pick the best construction for $D_{i,j}$ depending on the value $|i-j|$. If $|i-j| \leq n^{1/3}$, we employ the algebraic construction. It provides for each integer $d' \geq 3$ a depth-d' $\mathsf{AC}^0[\oplus]$ circuit of size at most $\exp(O(\log n \cdot |i-j|^{2/(d'-1)})) = \exp(\widetilde{O}\big(n^{\frac{2}{3}\cdot\frac{1}{(d'-1)}}\big))$. On the other hand, if $|i-j| > n^{1/3}$ the combinatorial construction gives for each integer $d' \geq 3$ a depth-d' AC^0 circuit of size at most $\exp(O(n/|i-j|)^{1/(d'-2)}) = \exp(O\big(n^{\frac{2}{3}\cdot\frac{1}{(d'-2)}}\big))$. Overall, we obtain a depth-d $\mathsf{AC}^0[\oplus]$ circuit for f_n of size at most $2^{\widetilde{O}\big(n^{\frac{2}{3}\cdot\frac{1}{(d-4)}}\big)}$. ◀

2.2 An upper bound for linear threshold functions

Recall that an *exact threshold function* is a boolean function $f(x_1,\ldots,x_n)$ that evaluates to 1 if and only if $\sum_i w_i x_i = t$, where $w_1,\ldots,w_n,t \in \mathbb{R}$.

▶ **Lemma 6.** *Any threshold function on n variables can be computed by a polynomial-size constant-depth circuit with unbounded fan-in AND and OR gates, and a single layer of exact threshold gates each of fan-in n.*

Proof. The lemma is implicit in the work of Hansen and Podolskii [13]. Indeed, the proof of Theorem 7 in their paper shows that every threshold function on n variables can be written as a polynomial-sized OR of exact threshold functions, each of which is also on n variables. ◀

▶ **Lemma 7.** *Any exact threshold function on n variables can be computed by a polynomial-size constant-depth circuit with unbounded fan-in AND and OR gates, and a single layer of symmetric gates each of fan-in n.*

Proof. Our proof proceeds via Chinese remaindering, which is a common technique in the study of threshold functions.

Suppose the exact threshold function is $\sum_i w_i x_i = t$, where we can assume w.l.o.g. that each w_i as well as t are integers that are $n^{O(n)}$ (we refer to [13] for more information about exact threshold gates). Let $p_1, p_2 \ldots p_{n^2}$ be the first n^2 primes. Using the upper bounds on each w_i, we have that for any input x, $\sum_i w_i x_i$ is characterized by its sequence of remainders modulo $\{p_j\}, j = 1 \ldots n^2$; the same holds for t. We design a circuit that is an AND of n^2 circuits C_j, one for each p_j, where circuit C_j verifies that $(\sum_i w_i x_i) \mod p_j = t \mod p_j$.

We now describe how to construct any fixed C_j. Note that $t \mod p_j$ is a fixed quantity independent of the input, so our task reduces to computing $(\sum_i w_i x_i) \mod p_j$ and taking an AND of the output bits or their negations as appropriate.

Let w_{ij} be $w_i \mod p_j$ for each $i \in [1, n], j \in [1, n^2]$. We need to compute $\sum_i w_{ij} x_i$ using a polynomial-size constant-depth circuit with unbounded fan-in AND and OR gates, and a single layer of symmetric gates each of fan-in n.

It follows from the Prime Number Theorem that each w_{ij} has at most $3\log(n)$ bits in its binary representation, for large enough n. We write each w_{ij} as $\sum_k w_{ijk} 2^k$, where w_{ijk} is the kth bit in the binary representation of w_{ij}, for $k \leq 3\log(n)$.

For each j and k, consider the following circuit B_{jk}. It has n inputs, where the i'th input bit is the AND of w_{ijk} (which is a fixed bit independent of the input) and x_i. B_{jk} computes the sum of these n inputs – this can be done by using at most $\lceil \log(n) \rceil$ symmetric gates in parallel, each of fan-in n.

Let y_{jk} be the output of each circuit B_{jk}. C_j computes $(\sum_k y_{jk} 2^k) \mod p_j$ using a constant-depth circuit of polynomial size. This can be done because there are only $O(\log(n))^2$ input bits and the function we are computing is in NC^1 (see e.g. [19]); it is folklore that any NC^1 function on polylogarithmically many bits can be computed by polynomial-size AC^0 circuits.

Summing up, our circuit has $\mathrm{poly}(n)$ size and $O(1)$ depth, and has a single layer of symmetric gates with fan-in n, as promised. ◀

▶ **Theorem 8.** *There is an integer c such that for every integer $d > c$, if $f_n : \{0,1\}^n \to \{0,1\}$ is a threshold function, then it can be computed by an $\mathsf{AC}^0[\oplus]$ circuit of depth d and of size $2^{\widetilde{O}(n^{\frac{2}{3} \cdot \frac{1}{(d-c)}})}$.*

Proof. We combine Lemmas 6 and 7 with Theorem 4. From Lemma 6 and Lemma 7, it follows that every threshold function on n variables can be computed by a polynomial-size constant-depth circuit with unbounded fan-in AND and OR gates, and a single layer of symmetric gates each of fan-in n. Suppose that the depth of this circuit is k. We set $c = k + 4$.

By using Theorem 4, we can replace each symmetric gate by a $\mathsf{AC}^0[\oplus]$ circuit of depth $d-k$ and of size $2^{\widetilde{O}(n^{\frac{2}{3}\cdot\frac{1}{(d-c)}})}$. The total size of the resulting circuit is $\mathrm{poly}(n)2^{\widetilde{O}(n^{\frac{2}{3}\cdot\frac{1}{(d-c)}})}$ and its depth is d. We can absorb the polynomial factor in the circuit size into the exponential term, to yield the result stated in the theorem. ◀

We have made no attempt to optimize the integer c in the statement of Theorem 8.

Note that the same proof as for Theorem 8 yields that any Boolean function truth-table reducible to linear threshold functions using a polynomial-size AC^0 reduction, where the size of any query is at most n, is also computable by $\mathsf{AC}^0[\oplus]$ circuits of the same size and depth as in the statement of the theorem. In particular, this is the case for polytopes, since any polytope over n variables is simply an AND of linear threshold functions over n variables.

2.3 A depth-4 upper bound

For any $n \geq 1$ and $i \in \{0,\ldots,n\}$, let $E_{n,i}$ denote the n-variable Boolean function that accepts inputs of Hamming weight i and rejects all other inputs.

▶ **Theorem 9.** *Any symmetric function on n variables has a depth-4 $\mathsf{AC}^0[\oplus]$ circuit of size $\exp(O(n^{1/4}\cdot(\log n)^{3/4}))$.*

This improves on the AC^0 upper bound of $\exp(\widetilde{O}(n^{1/3}))$ [8], which is tight up to log factors in the exponent [14]. To prove the above theorem, it suffices to show the following upper bound for exact majorities.

▶ **Lemma 10.** *Assume n is even. Then $E_{n,n/2}$ has a depth-4 $\mathsf{AC}^0[\oplus]$ circuit C of size $\exp(O(n^{1/4}\cdot(\log n)^{3/4}))$ with the output gate being an OR gate.*

We first prove Theorem 9 assuming Lemma 10.

Proof of Theorem 9. By Lemma 10, we have a depth-4 $\mathsf{AC}^0[\oplus]$ circuit C of size $\exp(O(n^{1/4}\cdot(\log n)^{3/4}))$ with an OR output gate that computes $E_{2n,n}$. Note that this yields a circuit C_i for $E_{n,i}$ via the substitution $C_i(x_1,\ldots,x_n) = C(x0^i1^{n-i})$; observe that C_i is also a depth-4 $\mathsf{AC}^0[\oplus]$ circuit of size $\exp(O(n^{1/4}(\log n)^{3/4}))$ with an OR output gate.

Since any symmetric function on n variables is an OR of a subset of the $E_{n,i}$, this yields the theorem. ◀

We now discuss the proof of Lemma 10. Let r, s be growing functions of n with $s = o(n/\log n)$. We will design a *random* depth-3 circuit $\boldsymbol{C}'_{n,r,s}$ such that

1. For any input a of Hamming weight $k \neq n/2$, $\Pr_{\boldsymbol{C}'_{n,r,s}}[\boldsymbol{C}'_{n,r,s}(a) = 1] = 0$.
2. For any input a of Hamming weight $n/2$, $\Pr_{\boldsymbol{C}'_{n,r,s}}[\boldsymbol{C}'_{n,r,s}(a) = 1] \geq p \stackrel{\text{def}}{=} n^{-r}$.

(The parameter s will be used to optimize the size of the final circuit.)

The construction of C will easily follow from that of $\boldsymbol{C}'_{n,r,s}$. The latter, which we now describe, uses a modification of Amano's construction [4] of random formulas for approximating the Majority function (which itself builds upon [2, 28, 21]), some basic facts about polynomial interpolation, and well-known ideas for computing Exact majorities [8, 22].

The lemma below is Amano's construction with a few parameters modified.

▶ **Lemma 11.** *Let m be a growing parameter and $\delta = o(1/\log m)$. There exists a random $\bigwedge\bigvee\bigwedge$ formula $\boldsymbol{F}_3$ of size $\exp(O(\sqrt{(\log m)/\delta}))$ such that*

1. *For any input a of Hamming weight $i \leq m((1/2)-\delta)$, $\Pr_{\boldsymbol{F}_3}[\boldsymbol{F}_3(a) = 1] = 0$.*
2. *For any input a of Hamming weight $i \geq m/2$, $\Pr_{\boldsymbol{F}_3}[\boldsymbol{F}_3(a) = 1] \geq (3/4)$.*

A proof sketch is given later in this section. The construction of the random circuit $\boldsymbol{C}'_{n,r,s}$ now proceeds as follows.

1. Divide the n input variables $x_1, \ldots, x_n$ randomly into r buckets $B_1, \ldots, B_r$ of size n/r each. We assume $r|n$ and that $m \stackrel{\text{def}}{=} n/r$ is even for simplicity.
2. Let $\delta = s/m$. For each bucket B_i, use Lemma 11 to construct a random $\bigwedge\bigvee\bigwedge$ formula $\boldsymbol{F}^{(i)}$ of size $\exp(O(\sqrt{(1/\delta)\log m}))$ on the variables in B_i that accepts no input of Hamming weight at most $m((1/2)-\delta)$ and accepts each input of Hamming weight at least $m/2$ with probability at least $3/4$.
   Define $\boldsymbol{G}^{(i)}$ to be $\boldsymbol{F}^{(i)}(\neg x : x \in B_i)$. Note that $\boldsymbol{G}^{(i)}$ accepts no input of Hamming weight at least $m((1/2)+\delta)$ and accepts each input of Hamming weight at most $m/2$ with probability at least $3/4$.
   Let $\boldsymbol{H}^{(i)} = \boldsymbol{F}^{(i)} \wedge \boldsymbol{G}^{(i)}$. By a union bound, $\boldsymbol{H}^{(i)}$ accepts each input of Hamming weight *exactly* $m/2$ with probability at least $1/2$ and no input of Hamming weight k such that $|k-(m/2)| \geq \delta m$.
3. For each bucket B_i, let $P^{(i)} \in \mathbb{F}_2[x : x \in B_i]$ be a multilinear polynomial of degree at most $2s$ that accepts inputs of Hamming weight $m/2$ but no input of Hamming weight k such that $|k-(m/2)| < s$. Such a polynomial exists by standard interpolation arguments (cf. Lemma 5; for a proof see e.g. Alman and Williams [3, Proof of Lemma 3.1]).
   We think of $P^{(i)}$ as a depth-2 $\bigoplus\bigwedge$ formula of size $n^{O(s)}$.
4. Finally, we define $\boldsymbol{C}'_{n,r,s} = \bigwedge_{i\in[r]} \left(\boldsymbol{H}^{(i)} \wedge P^{(i)}\right)$.
   By construction, $\boldsymbol{C}'_{n,r,s}$ is a depth-3 $\mathsf{AC}^0[\oplus]$ circuit of size $\mathrm{poly}(n) \cdot \exp(O(s\log n + \sqrt{(m/s)\log n}))$.
   Given any input a of Hamming weight $k \neq n/2$, there is a bucket B_i such that the restriction $a^{(i)}$ to B_i has weight $k_i \neq m/2$. In this case, either $\boldsymbol{H}^{(i)}$ or $P^{(i)}$ rejects $a^{(i)}$ (depending on whether $|k_i - m| \geq s$ or $|k_i - m| < s$ respectively). Hence, $\boldsymbol{C}'_{n,r,s}$ rejects a (with probability 1).
   Conversely, given an input a of Hamming weight $n/2$, $\boldsymbol{C}'_{n,r,s}$ accepts a if its restriction $a^{(i)}$ of a to each bucket B_i has weight exactly $m/2$ and we have a good choice for the randomness of each $\boldsymbol{H}^{(i)}$. The probability of this is at least

$$\left(\frac{3}{4}\right)^r \cdot \frac{\binom{m}{m/2}^r}{\binom{n}{n/2}} \geq \frac{(2^m/10\sqrt{m})^r}{2^n} \geq \frac{1}{n^r} = p.$$

So we have constructed $\boldsymbol{C}'_{n,r,s}$ as required. In order to convert this to a circuit for $E_{n,n/2}$, we use a standard covering argument. Let $t = n/p$. We choose independent random circuits $\boldsymbol{C}_1, \ldots, \boldsymbol{C}_t$ where each $\boldsymbol{C}_i$ has the same distribution as $\boldsymbol{C}'_{n,r,s}$. Define $\boldsymbol{C}_{n,r,s} = \bigvee_i \boldsymbol{C}_i$.

Clearly, $\boldsymbol{C}_{n,r,s}$ accepts no input a of Hamming weight $k \neq n/2$. On the other hand, the probability that $\boldsymbol{C}_{n,r,s}$ rejects an input a of weight $n/2$ can be upper bounded by $(1-p)^t \leq \exp(-pt) = \exp(-n)$. By a union bound, the probability that $\boldsymbol{C}_{n,r,s}$ rejects *some* input of weight $n/2$ is at most $\binom{n}{n/2} \cdot \exp(-n) < 1$.

In particular, by averaging, there is a fixed circuit $C_{n,r,s}$ in the support of the distribution of $\boldsymbol{C}_{n,r,s}$ that computes $E_{n,n/2}$ correctly on all inputs.

By construction, the circuit $C_{n,r,s}$ has size $\mathrm{poly}(n) \cdot \exp(O((r+s)\log n + \sqrt{(m/s)\log n}))$. Setting $r = s = \Theta((n/\log n)^{1/4})$, we get a circuit C of the claimed size. This completes the proof of Lemma 10.

Proof Sketch of Lemma 11. We provide a sketch of the proof, omitting calculations. The reader is invited to consult Amano's paper [4] for more details.

Set $\ell = \lceil \sqrt{(\log m)/\delta} \rceil$ and define the random formulas $\boldsymbol{F}_i$ ($i \in [3]$) of depth i as follows.

1. $\boldsymbol{F}_1$ is simply an AND of size ℓ, where each input is chosen i.u.a.r. from among the input variables $\{x_1, \ldots, x_m\}$.
2. $\boldsymbol{F}_2$ is an OR of $L \stackrel{\text{def}}{=} \lceil 2^\ell \cdot (100\ell \ln 2) \rceil$ independent copies of $\boldsymbol{F}_1$.
3. $\boldsymbol{F}'_3$ is an AND of $M \stackrel{\text{def}}{=} 2^{100\ell - 3}$ independent copies of $\boldsymbol{F}_2$.
4. We define $\boldsymbol{F}_3$ to be $\boldsymbol{F}'_3$ conditioned on the event that $\boldsymbol{F}'_3$ does not accept any input of Hamming weight $i \leq m((1/2) - \delta)$.

Clearly, $\boldsymbol{F}_3$ is a random formula of the required size.
To argue that $\boldsymbol{F}_3$ has the required input-output behaviour, we proceed as follows.

1. Say a is an input of Hamming weight at least $m/2$.
   a. A uniformly random co-ordinate of a is 1 with probability at least $1/2$. Hence, $\boldsymbol{F}_1(a) = 1$ with probability at least $p_1 \stackrel{\text{def}}{=} 2^{-\ell}$.
   b. Hence, the probability that $\boldsymbol{F}_2$ rejects a is at most $(1 - p_1)^L \leq p_2 \stackrel{\text{def}}{=} 2^{-100\ell}$.
   c. Therefore, the probability that $\boldsymbol{F}'_3$ rejects a is at most $Mp_2 \leq 1/8$.
2. Now assume a' is an input of Hamming weight at most $m((1/2) - \delta)$.
   a. A uniformly random co-ordinate of a' is 1 with probability at most $1/2 - \delta$. Hence, $\boldsymbol{F}_1(a') = 1$ with probability at most $q_1 \stackrel{\text{def}}{=} p_1 \cdot (1 - \delta\ell)$.
   b. Hence, the probability that $\boldsymbol{F}_2$ rejects a' is at least $(1-q_1)^L \geq q_2 \stackrel{\text{def}}{=} p_2 \cdot \exp(10\delta\ell^2 \ln 2) \geq p_2 \cdot m^{10}$.
   c. Therefore, the probability that $\boldsymbol{F}'_3$ accepts a' is at most $(1-q_2)^M \leq \exp(-M \cdot p_2 \cdot m^{10}) \leq \exp(-m^9)$.
3. Thus, the probability that $\boldsymbol{F}_3$ accepts an input a of Hamming weight at least $m/2$ is at least $(7/8) - 2^m \cdot \exp(-m^9) \geq (3/4)$. This concludes the proof.

3 The Lower Bounds

3.1 A refined analysis of approximate-degree bounds

The main theorem of this section is the following result.

▶ **Theorem 12.** *Fix any $d \geq 2$. Let C be a depth-d $\mathsf{AC}^0[\oplus]$ circuit computing the n-bit majority function Maj_n. Then, C has size at least $\exp(\Omega(n^{1/(2d-3)}))$.*

We follow the lower bound approach of Razborov [23], who showed that any small $\mathsf{AC}^0[\oplus]$ circuit C can be suitably approximated by a low-degree polynomial. This is proved by iteratively constructing low-degree polynomials for the OR and AND gates of C (parity gates are low-degree by definition, and hence trivial to approximate), and then composing the polynomials together to obtain a low-degree approximation to C. We follow a similar idea, but make the (crucial) observation that the approximations for the AND and OR gates are one-sided (on opposite sides). This means that the construction of Razborov is slightly better than normally advertised: the error is much lower on $C^{-1}(b)$ than $C^{-1}(1-b)$ for some $b \in \{0,1\}$.

▶ **Definition 13.** *Let $f : \{0,1\}^n \to \{0,1\}$ be any Boolean function. For parameters $\varepsilon_0, \varepsilon_1 \in (0,1)$, an $(\varepsilon_0, \varepsilon_1)$-*error Probabilistic polynomial *for f is a random multilinear polynomial $\boldsymbol{P}$ chosen from $\mathbb{F}_2[x_1, \ldots, x_n]$ such that for $b \in \{0,1\}$ and any $a \in f^{-1}(b)$,*

$$\Pr_{\boldsymbol{P}}[\boldsymbol{P}(a) \neq f(a)] \leq \varepsilon_b.$$

We say that $\boldsymbol{P}$ *has degree at most* d (*denoted* $\deg(\boldsymbol{P}) \leq d$) *if the underlying distribution is supported on multilinear polynomials of degree at most* d. *We define the* $(\varepsilon_0, \varepsilon_1)$-error probabilistic degree *of* f (*denoted* $\mathrm{pdeg}_{\varepsilon_0,\varepsilon_1}(f)$) *to be the least* d *such that there is a* $\boldsymbol{P}$ *as above of degree at most* d.

Typically, the above is stated for $\varepsilon_0 = \varepsilon_1$, but it will be important for us to track the errors on the 0 and 1 inputs of f separately. For example, it allows us to observe the following feature of a construction due to Razborov [23]. (See also Kopparty's lecture notes [16] for a proof.)

▶ **Lemma 14** (Razborov [23]). *Let* AND_m *and* OR_m *denote the AND and OR functions on* m *inputs respectively. Then, for any* $\varepsilon > 0$, $\mathrm{pdeg}_{\varepsilon,0}(\mathrm{AND}_m)$ *and* $\mathrm{pdeg}_{0,\varepsilon}(\mathrm{OR}_m)$ *are both at most* $\lceil \log(1/\varepsilon) \rceil$.

From this, we get the following corollary.

▶ **Corollary 15.** *Let* C *be an* $\mathsf{AC}^0[\oplus]$ *circuit of size* s *and depth* $d \geq 1$. *Then, for any* $c \geq 1$ *and large enough* s, *we have*

$$\min\{\mathrm{pdeg}_{(1/10),(1/s^c)}(C), \mathrm{pdeg}_{(1/s^c),(1/10)}(C)\} \leq O(c \log s)^{d-1}$$

where the $O(\cdot)$ *hides an absolute constant.*

Proof Sketch. We assume that the output gate of C is either a parity gate or an OR gate (the case of the AND gate is similar to the case of the OR gate).

For each non-output gate g in the circuit (viewed as a function of its input wires), we first construct a $(1/s^{c+2}, 1/s^{c+2})$-error probabilistic polynomial $\boldsymbol{P}_g$ of degree $O(c \log s)$ for g. Note that the existence of $\boldsymbol{P}_g$ is trivial if g is a parity gate (since the parity function is a polynomial of degree 1) and otherwise, Lemma 14 gives us such a probabilistic polynomial.

For the output gate g_0, we construct a $(0, 1/20)$-error probabilistic polynomial $\boldsymbol{P}_{g_0}$ of degree $O(1)$: again, this is trivial if g_0 is a parity gate and follows from Lemma 14 if g_0 is an OR gate.

Composing these polynomials together, we get a probabilistic polynomial $\boldsymbol{P}$ of degree $O(c \log s)^{d-1} \cdot O(1) = O(c \log s)^{d-1}$. Further for any input $a \in \{0,1\}^n$, we have $\boldsymbol{P}(a) = C(a)$ unless there is some gate g of C such that $\boldsymbol{P}_g$ does not simulate g faithfully on the corresponding setting of its inputs. For non-output gates, this probability is at most $1/s^{c+2}$. For the output gate, this probability is either 0 or at most $1/20$ depending on whether $a \in C^{-1}(0)$ or $C^{-1}(1)$ respectively. A union bound now implies that $\mathrm{pdeg}_{1/s^c,1/10}(C) = O(c \log s)^{d-1}$. ◀

The rest of the proof follows the lower bound of Smolensky [25] on the probabilistic degree of the Majority function. More precisely, we prove the following.

▶ **Lemma 16.** *Let* n *be a growing parameter. There exist absolute constants* $\alpha, \beta > 0$ *such that for all large enough* n *and all* $\varepsilon \in (1/2^{\alpha n}, \beta)$, *we have*

$$\min\{\mathrm{pdeg}_{1/10,\varepsilon}(\mathrm{Maj}_n), \mathrm{pdeg}_{\varepsilon,1/10}(\mathrm{Maj}_n)\} = \Omega(\sqrt{n \log(1/\varepsilon)}).$$

Smolensky[2] proved the above for $\mathrm{pdeg}_{\varepsilon,\varepsilon}(\mathrm{Maj}_n)$. Note that the above statement in conjunction with Corollary 15 immediately implies Theorem 12. Putting the upper bound in Corollary 15 for $c = 1$ together with the lower bound in Lemma 16, we get

$$O(\log s)^{d-1} \geq \Omega(\sqrt{n \log s}),$$

which yields $s = \exp(\Omega(n^{1/(2d-3)}))$.

To prove Lemma 16, we follow a "dual" version of Smolensky's proof that appears in a result of Kopparty and Srinivasan [17], which itself follows the closely-related ideas of Aspnes, Beigel, Furst and Rudich [5] and Green [12]. It is not clear that this dual reformulation is necessary for the proof below, but the language of this formulation makes it easier to use some other results from the literature in this context.

We start with the notion of a certifying polynomial for a Boolean function.

▶ **Definition 17** (Certifying polynomials). *A non-zero multilinear polynomial $R \in \mathbb{F}_2[x_1, \ldots, x_n]$ is a* certifying polynomial *for $f : \{0,1\}^n \to \{0,1\}$ if f is constant on* $\mathrm{Supp}(R) \stackrel{\text{def}}{=} \{a \in \mathbb{F}_2^n \mid R(a) \neq 0\}$.

The following is an easy corollary of standard properties of multilinear polynomials (see e.g. [17, Lemma 3.3]).

▶ **Fact 18.** *If R is a certifying polynomial for Maj_n, then $\deg(R) \geq \lceil n/2 \rceil$.*

We now return to Lemma 16. Let $\boldsymbol{P}$ be an $(\varepsilon, 1/10)$-error[3] probabilistic polynomial for Maj_n, and let us assume $\deg(\boldsymbol{P}) \leq d$. We need to lower bound d. By a union bound and averaging, we can find a (deterministic) polynomial $P \in \mathbb{F}_2[x_1, \ldots, x_n]$ of degree at most d such that

$$\Pr_{x \in \mathrm{Maj}_n^{-1}(0)}[P(x) \neq 0] \leq 2\varepsilon \quad \text{and} \quad \Pr_{x \in \mathrm{Maj}_n^{-1}(1)}[P(x) \neq 1] \leq \frac{1}{4}. \tag{1}$$

It suffices to lower bound $\deg(P)$. To do so, we show that there is a non-zero polynomial $Q \in \mathbb{F}_2[x_1, \ldots, x_n]$ of low degree such that Q vanishes on all points in $E_0 \stackrel{\text{def}}{=} \{x \in \mathrm{Maj}_n^{-1}(0) \mid P(x) \neq 0\}$. We then consider the multilinear polynomial $R = P \cdot Q$ (we use the identity $x_i^2 = x_i$ to ensure that R is multilinear). Note that R vanishes on all points of Hamming weight less than $n/2$: given a of weight less than $n/2$, either $a \in E_0$, in which case $Q(a) = 0$, or $a \notin E_0$, which implies that $P(a) = 0$. If we could argue that R is a non-zero polynomial, then it follows that R is a certifying polynomial for Maj_n and hence has degree at least $\lceil n/2 \rceil$ (by Fact 18). On the other hand, $\deg(R) \leq \deg(P) + \deg(Q)$ which implies a lower bound on $\deg(P)$.

The main part of the above argument is arguing the non-zeroness of R. To do this, we would like to show that there is a low-degree polynomial Q such that Q vanishes on E_0, but there is an $a \in \mathrm{Supp}(P)$ such that $Q(a) \neq 0$. To argue the existence of a suitable such Q, we use a result of Nie and Wang [20]. Informally, the result says that if a parameter D is chosen so that the number of multilinear monomials of degree at most D is much larger than $|E_0|$, then constraining a polynomial of degree at most D to be zero on E_0 does not constrain it at too many other points.

---

[2] Smolensky actually proves lower bounds for mod functions, which we don't consider here. However, it is clear that his proof works also for Majority. As far as we know, this proof first appeared in Szegedy's PhD thesis [27]. See [9] for a more recent exposition. A different proof, also due to Smolensky, appears in [26].

[3] A symmetric argument can be used to argue about $\mathrm{pdeg}_{1/10,\varepsilon}(\mathrm{Maj}_n)$.

To make this precise, we define the *degree-D Closure of E_0*, denoted $\mathrm{cl}_D(E_0)$, to be the set of all $a \in \mathbb{F}_2^n$ such that $Q(a) = 0$ for each Q of degree at most D such that Q vanishes at all points in E_0. Clearly, $\mathrm{cl}_D(E_0) \supseteq E_0$ but could potentially be much larger. The result of Nie and Wang [20] bounds the closure of small sets in $\mathbb{F}_2^n$ (see also the earlier results of Wei [29] and Keevash and Sudakov [15] which prove similar or stronger statements in different language).

▶ **Theorem 19** (Nie and Wang [20]). *Fix any $E \subseteq \mathbb{F}_2^n$ and any $D \geq 1$. Let N_D denote the number of multilinear monomials of degree at most D. Then, we have*

$$\frac{|\mathrm{cl}_D(E)|}{2^n} \leq \frac{|E|}{N_D}.$$

▶ Remark 20. Note that the above theorem generalizes the following standard fact (which follows easily from linear algebra): if $|E| < N_D$, then there is a non-zero polynomial of degree at most D that vanishes on E. Smolensky's proof (as formulated in [17]) can be seen as using only this special case of Theorem 19. Using the theorem in its full generality is what yields the stronger result below.

We are now ready to prove Lemma 16.

Proof of Lemma 16. It suffices to lower bound $\deg(P)$ where P is as in (1). Let E_0 be as defined above; by (1), we have $|E_0| \leq 2\varepsilon \cdot 2^n$. Also define $S = \mathrm{Supp}(P)$. Note that S contains all those $a \in \mathrm{Maj}_n^{-1}(1)$ such that $P(a) = 1$ which, by (1), has size at least $2^n \cdot (3/8 - o(1))$.

Now, choose the least D such that $N_D \geq (20\varepsilon) \cdot 2^n$. As long as α and β are small enough constants, we have $D = (n/2) - \Omega(\sqrt{n \log(1/\varepsilon)})$. Also, by Theorem 19, we know that $|\mathrm{cl}_D(E_0)| \leq 2^n/10 < |S|$. In particular, $S \nsubseteq \mathrm{cl}_D(E_0)$. This means that there is some $a_0 \in S$ and some Q of degree at most D such that Q vanishes on E_0 but not at a_0.

Let $R = P \cdot Q$ (we assume R is multilinear by using the identity $x_i^2 = x_i$). As argued above, R vanishes at all points in $\mathrm{Maj}_n^{-1}(0)$ and further, $R(a_0) = P(a_0)Q(a_0) \neq 0$. Hence, R is a non-zero polynomial such that Maj_n is the constant function 1 on inputs from $\mathrm{Supp}(R)$. By Fact 18, we have $\deg(R) \geq n/2$.

This implies that $\deg(P) \geq n/2 - D = \Omega(\sqrt{n \log(1/\varepsilon)})$. ◀

3.2 A depth-3 lower bound

In this section, we show how to use the ideas from the proof of Theorem 12 in conjunction with standard AC^0 lower bound techniques to get a near optimal lower bound of $\exp(\Omega(\sqrt{n}))$ for depth-3 circuits (there is an AC^0 circuit of size $\exp(\widetilde{O}(\sqrt{n}))$ computing the Majority function [8]).

▶ **Theorem 21.** *Let C be any depth-3 $\mathsf{AC}^0[\oplus]$ circuit computing the n-bit Majority function Maj_n. Then, C has size $\exp(\Omega(\sqrt{n}))$.*

The proof requires random restriction arguments [11, 1]. Recall that a *restriction* on n variables $x_1, \ldots, x_n$ is a function $\rho\colon \{x_1, \ldots, x_n\} \to \{0, 1, *\}$. A *Random restriction with $*$-probability $p \in [0, 1]$* is a random function $\boldsymbol{\rho}\colon \{x_1, \ldots, x_n\} \to \{0, 1, *\}$ where $\boldsymbol{\rho}^{-1}(*)$ is chosen to be a random subset $S \subseteq [n]$ of size $\lfloor pn \rfloor$ and each $\boldsymbol{\rho}(x_i)$ $(x_i \notin S)$ is set to 0 or 1 independently with probability $(1 - p)/2$ each. We use $\boldsymbol{\rho} \sim \mathcal{R}_p^n$ to denote the fact $\boldsymbol{\rho}$ is a random restriction on n variables with $*$-probability p.

Given a Boolean function $f : \{0, 1\}^n \to \{0, 1\}$, and a restriction $\rho : \{x_1, \ldots, x_n\} \to \{0, 1, *\}$, we use $f|_\rho$ to denote the restriction of f obtained by substituting variables as dictated by ρ (variables in $\rho^{-1}(*)$ are left as is).

We recall the Switching Lemma of Håstad [14] (this version is from Beame's survey [7]).

▶ **Lemma 22** (Håstad's Switching Lemma). *Let φ be a k-CNF or k-DNF. Then for $p \leq 1/10k$, we have*

$$\Pr_{\rho\sim\mathcal{R}_p}[\varphi|_\rho \text{ has no decision tree of depth at most } t] \leq (7pk)^t.$$

We say that a restriction ρ is *balanced* if $|\rho^{-1}(1)| = |\rho^{-1}(0)|$. Balanced restrictions will be useful for us since for a balanced restriction ρ, we have $\mathrm{Maj}_n|_\rho = \mathrm{Maj}_{|\rho^{-1}(*)|}$. Lemma 22 easily implies a similar corollary for random balanced restrictions.

▶ **Corollary 23.** *Let φ be a k-CNF or k-DNF on n variables. Then for $p \leq 1/10k$ such that $(n - \lfloor pn \rfloor)$ is even, we have*

$$\Pr_{\rho\sim\mathcal{R}^n_p}[\varphi|_\rho \text{ has no decision tree of depth at most } t \mid \rho \text{ is balanced}] \leq O(\sqrt{n}) \cdot (7pk)^t.$$

Proof. Follows directly from Lemma 22 and Bayes' rule since the probability that a random $\rho \sim \mathcal{R}^n_p$ is balanced is at least $\Omega(1/\sqrt{n})$. ◀

Using Corollary 23, we can derive the following simplification lemma for general depth-2 $\mathsf{AC}^0[\oplus]$ circuits.

▶ **Lemma 24.** *Let n, s be growing parameters with $s \geq n^2$. Let C' be any $\mathsf{AC}^0[\oplus]$ circuit on n variables of depth 2 and size at most s. Assume $p \leq 1/(500 \log s)$ is chosen so that $(n - \lfloor pn \rfloor)$ is even. Then, for large enough n, s, we have*

$$\Pr_{\boldsymbol{\rho}\sim\mathcal{R}^n_p}[\mathrm{pdeg}_{1/s^2,1/s^2}(C'|_{\boldsymbol{\rho}}) > 10 \log s \mid \boldsymbol{\rho} \text{ is balanced}] < \frac{1}{10s}.$$

Proof. The proof is a routine application of the switching lemma. We provide details for completeness.

To avoid some technicalities, we assume that n and $n/10$ are even integers. The proof can easily be extended to the other cases.

We use $\boldsymbol{\rho} \sim \mathcal{R}^n_{p,bal}$ to denote that ρ is a random restriction on n variables with $*$-probability p conditioned on being balanced. We sample $\boldsymbol{\rho}$ in two steps: we sample random restrictions $\boldsymbol{\rho}_1 \sim \mathcal{R}^n_{1/10,bal}$ and $\boldsymbol{\rho}_2 \sim \mathcal{R}^{n/10}_{10p,bal}$ and set $\boldsymbol{\rho}$ to be their composition $\boldsymbol{\rho}_2 \circ \boldsymbol{\rho}_1$ (i.e. we apply $\boldsymbol{\rho}_2$ to the variables in $\boldsymbol{\rho}_1^{-1}(*)$).

We first analyze the effect of applying $\boldsymbol{\rho}_1$. Consider any OR or AND gate g at depth 1 in C'. Say that g is *bad* for $\boldsymbol{\rho}_1$ if $g|_{\boldsymbol{\rho}_1}$ has fan-in at least $5 \log s$. Applying Corollary 23 with $k = 1$, we get for any gate g at depth 1,

$$\Pr_{\boldsymbol{\rho}_1}[g \text{ bad for } \boldsymbol{\rho}_1] \leq O(\sqrt{n}) \cdot (7/10)^{5 \log s} < 1/(20s^2)$$

for large enough s. Union bounding over all gates g at depth 1 (there are at most s of them), we see that with probability at least $1 - 1/(20s)$, all gates at depth 1 are good for $\boldsymbol{\rho}_1$. Condition on such a setting ρ_1 of the random restriction $\boldsymbol{\rho}_1$. By definition of ρ_1, the circuit $C'_1 \stackrel{\text{def}}{=} C'|_{\rho_1}$ has the property that all the AND and OR gates of depth 1 in C'_1 have fan-in at most $5 \log s$: in particular, they are polynomials of degree at most $5 \log s$.

Now we analyze the effect of $\boldsymbol{\rho}_2$ on C'_1. This is by a case analysis on the output gate g_0 of C'_1. Since the statement of the lemma is true for C' if and only if it is true for $\neg C'$, we can assume w.l.o.g. that the output gate of C' is either a parity gate or an OR gate.

1. **g_0 is a parity gate**: In this case, since the OR and AND gates at depth 1 compute polynomials of degree at most $5 \log s$, the entire circuit C'_1 already computes a polynomial of degree at most $5 \log s$. In particular, $C'_1|_{\boldsymbol{\rho}_2}$ has degree at most $5 \log s$ with probability 1.
2. **g_0 is an OR gate**: Then, we can write $C'_1 = C'_{1,1} \vee C'_{1,2}$, where $C'_{1,1}$ is an OR of parity gates and $C'_{1,2}$ is a $(5 \log s)$-DNF.
   $C'_{1,2}|_{\boldsymbol{\rho}_2}$ continues to be an OR of parities. By Lemma 14, any OR (and hence any OR of parities) has $(0, 1/s^2)$-probabilistic degree at most $\lceil 2 \log s \rceil \leq 3 \log s$. In particular, we have $\mathrm{pdeg}_{0,1/s^2}(C'_{1,2}|_{\boldsymbol{\rho}_2}) \leq 5 \log s$ with probability 1.
   For $C'_{1,1}$, we apply Corollary 23 (the Switching lemma) with $k = 5 \log s$. The random restriction $\boldsymbol{\rho}_2$ has $*$-probability $10p \leq 1/50 \log s = 1/10k$. Corollary 23 implies that the probability that $C'_{1,1}$ does not have a decision tree of height $5 \log s$ is at most $O(\sqrt{n}) \cdot (1/10)^{5 \log s} < 1/(20s)$. As a decision tree of height t can be represented as a polynomial of degree at most t, we see that with probability $1 - 1/(20s)$, the restricted $C'_{1,1}$ has degree at most $5 \log s$.
   Consequently, we see that with probability $1 - 1/(20s)$, we have both $\mathrm{pdeg}_{0,1/s^2}(C'_{1,2}|_{\boldsymbol{\rho}_2}) \leq 5 \log s$ and $\deg(C'_{1,1}) \leq 5 \log s$. When this happens, we also have $\mathrm{pdeg}_{0,1/s^2}(C'_1|_{\boldsymbol{\rho}_2}) \leq 10 \log s$ (the probabilistic polynomial for C'_1 can be obtained by composing the polynomial for the 2-bit OR function with polynomials for $C'_{1,1}$ and $C'_{1,2}$).

In both cases above, we have shown that

$$\Pr_{\boldsymbol{\rho}_2}[\mathrm{pdeg}_{(1/s^2,1/s^2)}(C'_1|_{\boldsymbol{\rho}_2}) > 10 \log s] < \frac{1}{20s}.$$

Along with our analysis of $\boldsymbol{\rho}_1$, this implies

$$\Pr_{\boldsymbol{\rho}}[\mathrm{pdeg}_{(1/s^2,1/s^2)}(C'|_{\boldsymbol{\rho}}) > 10 \log s] < \frac{1}{20s} + \frac{1}{20s} = \frac{1}{10s}. \qquad \blacktriangleleft$$

We are now ready to prove Theorem 21.

Proof of Theorem 21. Assume that C has size $s \leq \exp(\sqrt{n}/100)$, since otherwise we are done. Let $C'_1, \ldots, C'_s$ be the depth-2 subcircuits of C. Fix $p = \Theta(1/\log s)$ so that Lemma 24 is applicable.

Using Lemma 24 and applying a union bound over $i \in [s]$, we get

$$\Pr_{\boldsymbol{\rho} \sim \mathcal{R}^n_p}[\,\exists i \in [s],\ \mathrm{pdeg}_{1/s^2,1/s^2}(C'_i|_{\boldsymbol{\rho}}) > 10 \log s \mid \boldsymbol{\rho} \text{ is balanced}\,] < \frac{1}{10}.$$

In particular, there is a balanced restriction ρ on $\{x_1, \ldots, x_n\}$ such that $|\rho^{-1}(*)| = m = \Theta(n/\log s)$, and further, $\mathrm{pdeg}(C'_i|_\rho) \leq 10 \log s$ for each $i \in [s]$. Fix such a restriction ρ. W.l.o.g. we assume $\rho^{-1}(*) = \{x_1, \ldots, x_m\}$.

Fix $(1/s^2, 1/s^2)$-error probabilistic polynomials $\boldsymbol{P}_i(x_1, \ldots, x_m)$ of degree at most $10 \log s$ for C'_i ($i \in [s]$). We assume that the output gate g of C is either an OR gate or a parity gate (the case when the output is an AND gate is similar). In either case, Lemma 14 implies that g has a $(0, 1/10)$-error probabilistic polynomial $\boldsymbol{P}$ of constant degree.

Define $\boldsymbol{Q}(x_1, \ldots, x_m) = \boldsymbol{P}(\boldsymbol{P_1}, \ldots, \boldsymbol{P}_s)$. Clearly, $\deg(\boldsymbol{Q}) \leq O(\max_i \deg(\boldsymbol{P}_i)) = O(\log s)$. Also, it is easy to see that $\boldsymbol{Q}$ is a $(1/s, 1/5)$-error probabilistic polynomial for $C|_\rho = \mathrm{Maj}_n|_\rho = \mathrm{Maj}_m$. Lemma 16 therefore implies that $\deg(\boldsymbol{Q}) \geq \Omega(\sqrt{m \cdot \log s}) = \Omega(\sqrt{n})$, which implies that $s = \exp(\Omega(\sqrt{n}))$. $\blacktriangleleft$

3.3 An improved lower bound for all depths

In this section, we complete the proof of Theorem 2. The proof extends the ideas employed in the preceding sections in a natural way. The difference here is that the argument below employs the construction from Corollary 15 as an intermediate step, while the proof of Theorem 21 is slightly simpler and only requires Lemma 14.

Proof of Theorem 2. Let C be a depth-d $\mathsf{AC}^0[\oplus]$ circuit of size s that computes Majority over n input bits, where $d \geq 3$. Proceeding as in the proof of Theorem 21, we fix $p \stackrel{\text{def}}{=} \Theta(1/\log s)$, and apply Lemma 24 to the depth-2 subcircuits of C. By the same argument and after renaming input variables, this provides a balanced restriction ρ on $\{x_1, \ldots, x_m\}$ with $m \stackrel{\text{def}}{=} |\rho^{-1}(*)| = \Theta(n/\log s)$ and $(1/s^2, 1/s^2)$-error probabilistic polynomials $\boldsymbol{P}_i(x_1, \ldots, x_m)$ of degree $O(\log s)$ for each (ρ-restricted) depth-2 subcircuit C'_i of C.

We apply now the construction in Corollary 15 to the top $d-2$ layers of $C|_\rho$, replacing its depth-2 subcircuits by the probabilistic polynomials $\boldsymbol{P}_i(x_1, \ldots, x_m)$ obtained above. Adapting parameters in a straightforward way, this argument shows that $C|_\rho$ satisfies

$$\zeta \stackrel{\text{def}}{=} \min\{\mathrm{pdeg}_{(1/10),(1/s)}(C|_\rho), \mathrm{pdeg}_{(1/s),(1/10)}(C|_\rho)\} \leq O(c \log s)^{d-2}.$$

Moreover, since ρ is a balanced restriction the function computed by $C|_\rho$ is precisely Majority on m input bits.

We can assume w.l.o.g. that $s \leq 2^{\gamma\sqrt{n}}$ for a small enough (universal) constant $\gamma > 0$ independent of n and d. This allows us to invoke Lemma 16, which implies that $\zeta = \Omega(\sqrt{m \cdot \log s})$. Using the previously obtained upper bound on ζ and the value of m completes the proof of Theorem 2. ◀

References

1 Miklós Ajtai. $\sum^1_1$-formulae on finite structures. *Annals of Pure and Applied Logic*, 24(1):1–48, 1983.

2 Miklós Ajtai and Michael Ben-Or. A Theorem on Probabilistic Constant Depth Computations. In *Symposium on Theory of Computing* (STOC), pages 471–474, 1984.

3 Josh Alman and Ryan Williams. Probabilistic Polynomials and Hamming Nearest Neighbors. In *Symposium on Foundations of Computer Science* (FOCS), pages 136–150, 2015.

4 Kazuyuki Amano. Bounds on the Size of Small Depth Circuits for Approximating Majority. In *International Colloquium on Automata, Languages and Programming* (ICALP), pages 59–70, 2009.

5 James Aspnes, Richard Beigel, Merrick L. Furst, and Steven Rudich. The Expressive Power of Voting Polynomials. *Combinatorica*, 14(2):135–148, 1994.

6 David A. Mix Barrington, Richard Beigel, and Steven Rudich. Representing Boolean Functions as Polynomials Modulo Composite Numbers. *Computational Complexity*, 4:367–382, 1994. doi:10.1007/BF01263424.

7 Paul Beame. A switching lemma primer. Technical report, UW-CSE-95-07-01, 1994.

8 Ravi B. Boppana. Threshold Functions and Bounded Depth Monotone Circuits. In *Symposium on Theory of Computing* (STOC), pages 475–479, 1984. doi:10.1145/800057.808716.

9 Eshan Chattopadhyay, Pooya Hatami, Shachar Lovett, and Avishay Tal. Pseudorandom Generators from the Second Fourier Level and Applications to AC^0 with Parity Gates. In *Innovations in Theoretical Computer Science Conference* (ITCS), pages 22:1–22:15, 2019.

10 Xi Chen, Igor Carboni Oliveira, Rocco A. Servedio, and Li-Yang Tan. Near-optimal small-depth lower bounds for small distance connectivity. In *Symposium on Theory of Computing* (STOC), pages 612–625, 2016.

11 Merrick L. Furst, James B. Saxe, and Michael Sipser. Parity, Circuits, and the Polynomial-Time Hierarchy. *Mathematical Systems Theory*, 17(1):13–27, 1984.

12 Frederic Green. A complex-number Fourier technique for lower bounds on the Mod-m degree. *Computational Complexity*, 9(1):16–38, 2000.

13 Kristoffer Hansen and Vladimir Podolskii. Exact Threshold Circuits. In *Conference on Computational Complexity* (CCC), pages 270–279, 2010.

14 Johan Håstad. Almost Optimal Lower Bounds for Small Depth Circuits. In *Symposium on Theory of Computing* (STOC), pages 6–20, 1986.

15 Peter Keevash and Benny Sudakov. Set Systems with Restricted Cross-Intersections and the Minimum Rank of Inclusion Matrices. *SIAM J. Discrete Math.*, 18(4):713–727, 2005.

16 Swastik Kopparty. AC^0 lower bounds and pseudorandomness, 2013. Lecture notes on 'Topics in Complexity Theory and Pseudorandomness'. Can be found at: `http://sites.math.rutgers.edu/~sk1233/courses/topics-S13/lec4.pdf`.

17 Swastik Kopparty and Srikanth Srinivasan. Certifying Polynomials for $\mathrm{AC}^0[\oplus]$ Circuits, with Applications to Lower Bounds and Circuit Compression. *Theory of Computing*, 14(1):1–24, 2018.

18 Nutan Limaye, Karteek Sreenivasiah, Srikanth Srinivasan, Utkarsh Tripathi, and S. Venkitesh. The Coin Problem in Constant Depth: Sample Complexity and Parity gates. *Electronic Colloquium on Computational Complexity* (ECCC), 157, 2018.

19 Alexis Maciel and Denis Thérien. Threshold Circuits of Small Majority-Depth. *Inf. Comput.*, 146(1):55–83, 1998. `doi:10.1006/inco.1998.2732`.

20 Zipei Nie and Anthony Y Wang. Hilbert functions and the finite degree Zariski closure in finite field combinatorial geometry. *Journal of Combinatorial Theory, Series A*, 134:196–220, 2015.

21 Ryan O'Donnell and Karl Wimmer. Approximation by DNF: Examples and Counterexamples. In *International Colloquium on Automata, Languages and Programming* (ICALP), pages 195–206, 2007.

22 Ramamohan Paturi, Michael E. Saks, and Francis Zane. Exponential lower bounds for depth three Boolean circuits. *Computational Complexity*, 9(1):1–15, 2000. `doi:10.1007/PL00001598`.

23 Alexander A. Razborov. Lower bounds on the size of constant-depth networks over a complete basis with logical addition. *Mathematicheskie Zametki*, 41(4):598–607, 1987.

24 Benjamin Rossman and Srikanth Srinivasan. Separation of $\mathrm{AC}^0[\oplus]$ Formulas and Circuits. In *International Colloquium on Automata, Languages, and Programming* (ICALP), pages 50:1–50:13, 2017.

25 Roman Smolensky. Algebraic Methods in the Theory of Lower Bounds for Boolean Circuit Complexity. In *Symposium on Theory of Computing* (STOC), pages 77–82, 1987.

26 Roman Smolensky. On Representations by Low-Degree Polynomials. In *Symposium on Foundations of Computer Science* (FOCS), pages 130–138, 1993.

27 Mario Szegedy. *Algebraic Methods in Lower Bounds for Computational Models*. PhD thesis, University of Chicago, 1989.

28 Leslie G. Valiant. Short Monotone Formulae for the Majority Function. *J. Algorithms*, 5(3):363–366, 1984.

29 Victor K. Wei. Generalized Hamming weights for linear codes. *IEEE Transactions on Information Theory*, 37(5):1412–1418, 1991.

Size-Degree Trade-Offs for Sums-of-Squares and Positivstellensatz Proofs

Albert Atserias
Universitat Politècnica de Catalunya, Barcelona, Spain

Tuomas Hakoniemi
Universitat Politècnica de Catalunya, Barcelona, Spain

Abstract

We show that if a system of degree-k polynomial constraints on n Boolean variables has a Sums-of-Squares (SOS) proof of unsatisfiability with at most s many monomials, then it also has one whose degree is of the order of the square root of $n \log s$ plus k. A similar statement holds for the more general Positivstellensatz (PS) proofs. This establishes size-degree trade-offs for SOS and PS that match their analogues for weaker proof systems such as Resolution, Polynomial Calculus, and the proof systems for the LP and SDP hierarchies of Lovász and Schrijver. As a corollary to this, and to the known degree lower bounds, we get optimal integrality gaps for exponential size SOS proofs for sparse random instances of the standard NP-hard constraint optimization problems. We also get exponential size SOS lower bounds for Tseitin and Knapsack formulas. The proof of our main result relies on a zero-gap duality theorem for pre-ordered vector spaces that admit an order unit, whose specialization to PS and SOS may be of independent interest.

2012 ACM Subject Classification Theory of computation → Proof complexity

Keywords and phrases Proof complexity, semialgebraic proof systems, Sums-of-Squares, Positivstellensatz, trade-offs, lower bounds, monomial size, degree

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.24

Funding Both authors were partially funded by European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme, grant agreement ERC-2014-CoG 648276 (AUTAR) and MICCIN grant TIN2016-76573-C2-1P (TASSAT3).

Acknowledgements We are grateful to Michal Garlik, Moritz Müller and Aaron Potechin for comments on an earlier version of this paper. We are also grateful to Jakob Nordström for initiating a discussion on the several variants of the definition of *monomial size* as discussed in Section 2.

1 Introduction

A key result in semialgebraic geometry is the Positivstellensatz [33, 20], whose weak form gives a version of the Nullstellensatz for semialgebraic sets: A system of polynomial equations $p_1 = 0, \ldots, p_m = 0$ and polynomial inequalities $q_1 \geq 0, \ldots, q_\ell \geq 0$ on n commuting variables $x_1, \ldots, x_n$ has no solution over reals if and only if

$$-1 = s_\emptyset + \sum_{\substack{J \subseteq [\ell] \\ J \neq \emptyset}} s_J \prod_{j \in J} q_j + \sum_{j \in [m]} t_j p_j, \tag{1}$$

where the s_J are sums of squares of polynomials, and the t_j are arbitrary polynomials. Based on this, Grigoriev and Vorobjov [16] defined the Positivstellensatz (PS) proof system for certifying the unsatisfiability of systems of polynomial inequalities, and initiated the study of its proof complexity.

For most cases of interest, the statement of the Positivstellensatz stays true even if the first sum in (1) ranges only over singleton sets [31]. This special case of PS yields a proof system called Sums-of-Squares (SOS). Starting with the work in [3], SOS has received a good

deal of attention for its applications in algorithms and complexity theory. For the former, through the connection with the hierarchies of SDP relaxations [21, 27, 26, 8]. For the latter, through the lower bounds on the sizes of SDP lifts of combinatorial polytopes [11, 24, 23]. We refer the reader to the introduction of [26] for a discussion on the history of these proof systems and their relevance for combinatorial optimization.

In this paper we concentrate on the proof complexity of PS and SOS when their variables range over the Boolean hypercube, i.e., the variables come in pairs of twin variables x_i and $\bar{x}_i$, and are restricted through the axioms $x_i^2 - x_i = 0$, $\bar{x}_i^2 - \bar{x}_i = 0$ and $x_i + \bar{x}_i - 1 = 0$. This case is most relevant in combinatorial contexts. It is also the starting point for a direct link with the traditional proof systems for propositional logic, such as Resolution, through the realization that monomials represent Boolean disjunctions, i.e., clauses. In return, this link brings concepts and methods from the area of propositional proof complexity to the study of PS and SOS proofs.

In analogy with the celebrated size-width trade-off for Resolution [6] or the size-degree trade-off for Polynomial Calculus [17], a question that is suggested by this link is whether the *monomial size* of a PS proof can be traded for its *degree*. For a proof as in (1), the monomial size of the proof is the number of monomials in an explicit representation of the summands of the right-hand side. The degree of the proof is the maximum of the degrees of those summands. These are the two most natural measures of complexity for PS proofs (and precise definitions for both these measures will be made in Section 2). The importance of the question whether size can be traded for degree stems from the fact that, at the time of writing, the complexity of PS and SOS proofs is relatively well understood when it is measured by degree, but rather poorly understood when it is measured by monomial size. If size could be traded for degree, then strong lower bounds on degree would transfer to strong lower bounds on monomial size. The converse, namely that strong lower bounds on monomial size transfer to strong lower bounds on degree, has long been known by elementary linear algebra.

In this paper we answer the size-degree trade-off question for SOS, and for PS proofs of bounded *product width*, i.e., the number of inequalities that are multiplied together in (1). We show that if a system of degree-k polynomial constraints on n pairs of twin variables has a PS proof of unsatisfiability of product width w and no more than s many monomials in total, then it also has one of degree $O(\sqrt{n \log s} + kw)$. By taking $w = 1$, this yields a size-degree trade-off for SOS as a special case.

Our result matches its analogues for weaker proof systems that were considered before. Building on the work of [5] and [9], a size-width trade-off theorem was established for Resolution: a proof with s many clauses can be converted into one in which all clauses have size $O(\sqrt{n \log s} + k)$, where k is the size of the largest initial clause [6]. The same type of trade-off was later established for monomial size and degree for the Polynomial Calculus (PC) in [17], and for proof length and rank for LS and LS$^+$ [29], i.e., the proof systems that come out of the Lovász-Schrijver LP and SDP hierarchies [25]. To date, the question for PS and SOS had remained open, and is answered here[1].

Our proof of the trade-off theorem for PS follows the standard pattern of such previous proofs with one new key ingredient. Suppose Q is a system of equations and inequalities that has a size s refutation. Going back to the main idea from [9], the argument for getting

[1] Besides the proofs of the trade-off results for LS and LS$^+$, the conference version of [29] claims the result for the stronger Sherali-Adams and Lasserre/SOS proof systems, but the claim is made without proof. The very last section of the journal version [29] includes a sketch of a proof that, unfortunately, is an oversimplification of the LS/LS$^+$ argument that cannot be turned into a correct proof. The forthcoming discussion clarifies how our proof is based on, and generalizes, the one for LS/LS$^+$ in [29].

a degree d refutation goes in four steps: (1) find a variable x that appears in many large monomials, (2) set it to a value $b \in \{0,1\}$ to kill all monomials where it appears, (3) induct on the number of variables to get refutations of $Q[x=b]$ and $Q[x=\bar{b}]$ which, if s is small enough, are of degrees $d-1$ and d, respectively, and (4) compose these refutations together to get a degree d refutation of Q. The main difficulty in making this work for PS is step (4), for two reasons.

The first difficulty is that, unlike Resolution and the other proof systems, whose proofs are *deductive*, the proofs of PS are *formal identities*, also known as *static*. This means that, for PS, the reasoning it takes to refute Q from the degree $d-1$ refutation of $Q[x=b]$ and the degree d refutation of $Q[x=\bar{b}]$ needs to be witnessed through a single polynomial identity, without exceeding the bound d on the degree. This is challenging because the general simulation of a deductive proof by a static one incurs a degree loss. The second difficulty comes from the fact that, for establishing this identity, one needs to use a duality theorem that is not obviously available for degree-bounded PS proofs. What is needed is a zero-gap duality theorem for PS proofs of non-negativity that, in addition, holds tight at *each* fixed degree d of proofs. For SOS, the desired zero-gap duals are provided by the levels of the Lasserre hierarchy. This was established in [19] under the sole assumption that the inequalities include a ball contraint $B^2 - \sum_{i=1}^{n} x_i^2 \geq 0$ for some $B \in \mathbb{R}$. In the Boolean hypercube case, this can be assumed without loss of generality. For PS, we are not aware of any published result that establishes what we need, so we provide our own proof. At any rate, one of our contributions is the observation that a zero-gap duality theorem for PS-degree is a key tool for completing the step (4) in the proof of the trade-off theorem. We reached this conclusion from trying to generalize the proofs for LS and LS_+ from [29] to SOS. In those proofs, the corresponding zero-gap duality theorems are required only for the very special case where $d=2$ and for deriving linear inequalities from linear constraints. The fact that these hold goes back to the work of Lovász and Schrijver [25].

In the end, the zero-gap duality theorem for PS-degree turned out to follow from very general results in the theory of ordered vector spaces. Using a result from [28] that whenever a pre-ordered vector space has an order-unit a zero-gap duality holds, we are able to establish the following general fact: for any convex cone $\mathcal{C}$ of provably non-negative polynomials and its restriction $\mathcal{C}_{2d}$ to proofs of some even degree $2d$, if the ball constraints $R - x^2 \geq 0$ belong to $\mathcal{C}_2$ for all variables x and some $R \geq 0$, then a zero-gap duality holds for $\mathcal{C}_{2d}$ in the sense that

$$\sup\{r \in \mathbb{R} : p - r \in \mathcal{C}_{2d}\} = \inf\{E(p) : E \in \mathcal{E}_{2d}\},$$

where $\mathcal{E}_{2d}$ is an appropriate dual space for $\mathcal{C}_{2d}$. The conditions are easily seen to hold for PS-degree and SOS-degree in the Boolean hypercube case, and we have what we want. We use this in Section 3, where we prove the trade-off lemma, but defer its proof to Section 5.

In Section 4 we list some of the applications of the size-degree trade-off for PS that follow from known degree lower bounds. Among these we include exponential size SOS lower bounds for Tseitin formulas, Knapsack formulas, and optimal integrality gaps for sparse random instances of MAX-3-XOR and MAX-3-SAT. Except for Knapsack formulas, for which size lower bounds follow from an easy random restriction argument applied to the degree lower bounds in [13, 15], these size lower bounds for SOS appear to be new.

2 Preliminaries

For a natural number n we use the notation $[n]$ for the set $\{1,\ldots,n\}$. We write $\mathbb{R}_{\geq 0}$ and $\mathbb{R}_{>0}$ for the sets of non-negative and positive reals, respectively and $\mathbb{N}$ for the set of natural numbers. The natural logarithm is denoted log, and exp denotes base e exponentiation.

2.1 Polynomials and the Boolean ideal

Let $x_1, \ldots, x_n$ and $\bar{x}_1, \ldots, \bar{x}_n$ be two disjoint sets of variables. Each $x_i, \bar{x}_i$ is called a pair of twin variables, where x_i is the basic variable and $\bar{x}_i$ is its twin. We consider polynomials over the ring of polynomials with real coefficients and commuting variables $\{x_i, \bar{x}_i : i \in [n]\}$, which we write simply as $\mathbb{R}[x]$. The intention is that all the variables range over the Boolean domain $\{0, 1\}$, and that $\bar{x}_i = 1 - x_i$. Accordingly, let I_n be the Boolean ideal, i.e., the ideal of polynomials generated by the following set of *Boolean axioms* on the n pairs of twin variables:

$$B_n = \{x_i^2 - x_i : i \in [n]\} \cup \{\bar{x}_i^2 - \bar{x}_i : i \in [n]\} \cup \{x_i + \bar{x}_i - 1 : i \in [n]\}$$

We write $p \equiv q \mod I_n$ if $p - q$ is in I_n.

A monomial is a product of variables. A term is the product of a non-zero real and a monomial. A polynomial is a sum of terms. For $\alpha \in \mathbb{N}^{2n}$, we write x^α for the monomial $\prod_{i=1}^n x_i^{\alpha_i} \bar{x}_i^{\alpha_{n+i}}$, so polynomials take the form $\sum_{\alpha \in I} a_\alpha x^\alpha$ for some finite $I \subseteq \mathbb{N}^{2n}$. The *monomial size* of a polynomial p is the number of terms, and is denoted $\text{size}(p)$. A *sum-of-squares polynomial* is a polynomial of the form $s = \sum_{i=1}^k r_i^2$, where each r_i is a polynomial in $\mathbb{R}[x]$. For a polynomial $p \in \mathbb{R}[x]$ we write $\deg(p)$ for its degree. We think of $\mathbb{R}[x]$ as an infinite dimensional vector space, and we write $\mathbb{R}[x]_d$ for the subspace of polynomials of degree at most d.

2.2 Sums-of-Squares proofs

Let $Q = \{q_1, \ldots, q_\ell, p_1, \ldots, p_m\}$ be an indexed set of polynomials. We think of the q_j polynomials as inequality constraints, and of the p_j polynomials as equality constraints:

$$q_1 \geq 0, \ldots, q_\ell \geq 0, \quad p_1 = 0, \ldots, p_m = 0. \tag{2}$$

Let p be another polynomial. A *Sums-of-Squares (SOS) proof* of $p \geq 0$ from Q is a formal identity of the form

$$p = s_0 + \sum_{j \in [\ell]} s_j q_j + \sum_{j \in [m]} t_j p_j + \sum_{q \in B_n} u_q q, \tag{3}$$

where s_0 and $s_1, \ldots, s_\ell$ are sums of squares of polynomials, $s_j = \sum_{i=1}^{k_j} r_{i,j}^2$ for $j \in [\ell] \cup \{0\}$, and $t_1, \ldots, t_m$ and all u_q are arbitrary polynomials. The proof is of *degree at most* d if $\deg(p) \leq d$, $\deg(s_0) \leq d$, $\deg(s_j) + \deg(q_j) \leq d$ for each $j \in [\ell]$, and $\deg(t_j) + \deg(p_j) \leq d$ for each $j \in [m]$. The proof is of *monomial size at most* s if

$$\sum_{i=1}^{k_0} \text{size}(r_{i,0}) + \sum_{j \in [\ell]} \sum_{i=1}^{k_j} \text{size}(r_{i,j}) + \sum_{j \in [m]} \text{size}(t_j) \leq s.$$

This definition of size corresponds to the number of monomials of an explicit SOS proof given in the form $(s_0, s_1, \ldots, s_\ell, t_1, \ldots, t_m)$, where each s_j is given in the form $(r_{1,j}, \ldots, r_{k_j,j})$, and all the $r_{i,j}$ and t_j polynomials are represented as explicit sums of terms. Accordingly, the monomials of the $r_{i,j}$'s and the t_j's are called the *explicit monomials* of the proof.

Note that the u_q polynomials are not considered in the definition we have chosen of an explicit SOS proof, so they do not contribute to its monomial size or its degree. The rationale for this is that typically one thinks of the identity in (3) as an equivalence

$$p \equiv s_0 + \sum_{j \in [\ell]} s_j q_j + \sum_{j \in [m]} t_j p_j \mod I_n$$

and we want proof size and degree to not depend on how the computations modulo the Boolean ideal I_n are performed. For degree this choice is further justified from the fact that one may always assume that the degrees of the products $u_q q$ do not surpass the degree d in a proof of degree d. This follows from the fact that B_n is a Gröbner basis for I_n with respect to any monomial ordering – one can see this quite easily using Buchberger's Criterion (see e.g. [10]). In particular upper and lower bounds for the restricted definition of degree imply the same upper and lower bounds for our liberal definition of degree, and vice versa. For monomial size, this goes only in one direction: lower bounds on our liberal definition of monomial size translate into lower bounds for a restricted definition of monomial size that takes $\sum_{q\in B_n} \text{size}(u_q)$ also into account. Since our aim is to prove lower bounds on the number of monomials in a proof, proving our results for our more liberal definition of monomial size makes our results only stronger.

2.3 Positivstellensatz proofs

This proof system is an extension of SOS. Let $Q = \{q_1, \ldots, q_\ell, p_1, \ldots, p_m\}$ be an indexed set of polynomials interpreted as in (2). A *Positivstellensatz proof* (PS) of $p \geq 0$ from Q is a formal identity of the form

$$p = s_\emptyset + \sum_{J\in\mathcal{J}} s_J \prod_{j\in J} q_j + \sum_{j\in[m]} t_j p_j + \sum_{q\in B_n} u_q q, \tag{4}$$

where $\mathcal{J}$ is a collection of non-empty subsets of $[\ell]$, each s_J is a sum-of-squares polynomial, $s_J = \sum_{i=1}^{k_J} r_{i,J}^2$, and each t_j and u_q is an arbitrary polynomial. The proof is of *degree at most* d if $\deg(p) \leq d$, $\deg(s_\emptyset) \leq d$, $\deg(s_J)+\sum_{j\in J}\deg(q_j) \leq d$ for each $J \in \mathcal{J}$, and $\deg(t_j)+\deg(p_j) \leq d$ for each $j \in [m]$. The proof is of *monomial size at most* s if

$$\sum_{i=1}^{k_0} \text{size}(r_{i,\emptyset}) + \sum_{J\in\mathcal{J}} \sum_{i=1}^{k_J} \text{size}(r_{i,J}) + \sum_{j\in[m]} \text{size}(t_j) \leq s.$$

The proof has *product-width* at most w if each $J \in \mathcal{J}$ has cardinality at most w. The *explicit monomials* of the proof are the monomials of the $r_{i,J}$'s and the t_j's. It should be noted that PS applied to a Q that contains at most one inequality constraint (i.e., $\ell \leq 1$) is literally equivalent to SOS: any power of a single inequality is either a square, or the lift of that inequality by a square.

As in SOS proofs, the definitions of monomial size and degree of a proof do not take into account the u_q polynomials. Likewise, the monomials in the products $\prod_{j\in J} q_j$ do not contribute to the definition of monomial size. As above, this liberal definition plays in favour of lower bounds in the case of monomial size. For degree, ignoring the u_q's does not really matter, again, because B_n is a Gröbner basis for I_n.

2.4 More on the definition of monomial size

Starting at [9, 1], counting monomials in algebraic proof systems such as the Polynomial Calculus (PC) is a well-established practice in propositional proof complexity. One motivation for it comes from the fact that PC with twin variables, called PCR in [1], polynomially simulates Resolution, and the natural transformation that is given by the proof turns the clauses of the Resolution proof into monomials. Another motivation comes from the fact that, in the area of computational algebra, the performance of the Gröbner basis method appears to depend significantly on how the polynomials are represented. In this respect, the sum of

monomials representation of polynomials features among the first and most natural choices to be used in practice. That said, for the natural static version of PC called Nullstellensatz (NS) [4], let alone for SOS and PS, counting monomials does not appear to have such a well-established tradition. Note that in the presence of twin variables, SOS monomial size is known to polynomially simulate Resolution (see Lemma 4.6 in [2], where this is proved with a slightly different definition of SOS and monomial size from the one above; the difference is minor). It follows that the first of the two motivations for counting monomials in PC carries over to SOS, and hence to PS.

In the original Beame et al. and Grigoriev-Vorobjov papers [4, 16] where NS and PS were defined first, size is never considered, only degree. The subsequent Grigoriev's papers on SOS [13, 14] did not consider size either. To the best of our knowledge, the first reference that defines a notion of size for (the version of) PS proofs (with $w = 0$) appears to be [15], where the size of a proof is defined as "the length of a reasonable bit representation of all polynomials" in the proof. The same paper proves lower bounds on the "number of monomials" of an SOS proof (see Lemma 9.1 in [15]) without being precise as to whether it is counting monomials in the $r_{i,0}$ polynomials (in the notation of (3)), or in the expansion of s_0 as a sum of terms. Note, however, that $\text{size}(s_0) \leq \sum_i \text{size}(r_{i,0})^2$, hence the difference between these two possibilities is not terribly critical. As with the squares s_j, the definitions in [15] are not explicit as to whether the monomials in the t_j polynomials (in the notation of (3) again) contribute to the monomial size by themselves, or whether one is to take into account the expansions of the products $t_j p_j$. Unlike ours, the definitions in [15] do not distinguish between the u_q polynomials that multiply the Boolean axioms and the rest.

The difference between counting the monomials of the s_j (or the $r_{i,j}$) polynomials versus counting those in the expansions of the products $s_j q_j$ and $t_j p_j$ is again not critical if one is satisfied with a notion of size *up to* a polynomial factor that depends on the size of the input. If one is to care about such refinements of monomial size that take into account polynomial factors, then a natural size measure for, say, $t_j p_j$ could well be $\text{size}(t_j) + \text{size}(p_j)$ or even $\text{size}(t_j) \cdot \text{size}(p_j)$, instead of $\text{size}(t_j p_j)$. Note that $\text{size}(t_j) \cdot \text{size}(p_j)$ corresponds to the number of monomials that one would encounter while expanding the product $t_j p_j$ in the naive way *before* merging terms with the same monomial, and in particular, before any potential cancelling of terms occurs. In [2], the monomial size of (their slightly different version of) Lasserre/SOS is defined in terms of the expanded summands, which in the notation of (3), would correspond to $\text{size}(s_0) + \sum_j \text{size}(s_j q_j) + \sum_j \text{size}(t_j p_j) + \sum_q \text{size}(u_q q)$. In [22] the same convention for defining monomial size is used but the last sum over q is omitted since they work mod I_n by default. For PS proofs as in (4) that have large product-width w, whether we count the monomials in the s_J polynomials or in the expansions of the products $s_J \prod_{j \in J} q_j$ could make a significant difference, i.e., exponential in w. If we think of the proof in (4) as given by the indexed sequences $(s_J : J \in \mathcal{J} \cup \{\emptyset\})$ and $(t_j : j \in [m])$, then counting only the monomials in the s_J polynomial, or even better in the $r_{i,J}$ polynomials, looks like the natural choice.

3 Size-Degree Trade-Off

In this section we prove the following.

▶ **Theorem 1.** *For every two natural numbers n and k, every indexed set Q of polynomials of degree at most k with n pairs of twin variables, and every two positive integers s and w, if there is a PS refutation from Q of product-width at most w and monomial size at most s, then there is a PS refutation from Q of product-width at most w and degree at most $4\sqrt{2(n+1)\log(s)} + kw + 4$.*

An immediate consequence is a degree criterion for size lower bounds:

▶ **Corollary 2.** *Let Q be an indexed set of polynomials of degree at most k with n pairs of twin variables, and let w be a positive integer. If d is the minimum degree and s is the minimum monomial size of PS refutations from Q of product-width at most w, and $d \geq kw + 4$, then $s \geq \exp((d - kw - 4)^2/(32(n + 1)))$.*

The proof of Theorem 1 will follow the standard structure of proofs for degree-reduction lemmas for other proof systems, except for some complications in the *unrestricting* lemmas. These difficulties come from the fact that PS proofs are static. The main tool around these difficulties is a tight Duality Theorem for degree-bounded proofs with respect to so-called *cut-off functions* as defined next.

3.1 Duality modulo cut-off functions

Let $Q = \{q_1, \ldots, q_\ell, p_1, \ldots, p_m\}$ be an indexed set of polynomials interpreted as constraints as in (2). A *cut-off function* for Q is a function $c\colon \mathcal{P}([\ell]) \dot\cup [m] \to \mathbb{N}$ with $c(J) \geq \sum_{j\in J} \deg(q_j)$ for each $J \subseteq [\ell]$, and $c(j) \geq \deg(p_j)$ for each $j \in [m]$. A PS proof as in (4) has *degree mod c* at most d if $\deg(p) \leq d$, $\deg(s_0) \leq d$, $\deg(s_J) \leq d - c(J)$ for each $J \in \mathcal{J}$, and $\deg(t_j) \leq d - c(j)$ for each $j \in [m]$.

Let $\mathrm{PS}^c_{w,d}(Q)$ denote the set of all polynomials q of degree at most d such that $q \geq 0$ has a PS proof from Q of degree mod c at most d and product-width at most w. We write $Q \vdash^c_{w,d} q \geq p$ if $q - p \in \mathrm{PS}^c_{w,d}(Q)$. A *pseudo-expectation* for Q of degree mod c at most d and product-width at most w is a linear functional E from the set of all polynomials of degree at most d such that $E(1) = 1$ and $E(q) \geq 0$ for all $q \in \mathrm{PS}^c_{w,d}(Q)$. We denote by $\mathcal{E}^c_{w,d}(Q)$ the set of pseudo-expectations for the indicated parameters.

▶ **Theorem 3.** *Let d be a positive integer, let Q be an indexed set of polynomials, let c be a cut-off function for Q, let w be a positive integer, and let p be a polynomial of degree at most $2d$. Then*

$$\sup\{r \in \mathbb{R} : Q \vdash^c_{w,2d} p \geq r\} = \inf\{E(p) : E \in \mathcal{E}^c_{w,2d}(Q)\}.$$

Moreover, if the set $\mathcal{E}^c_{w,2d}(Q)$ is non-empty, then there is a pseudo-expectation achieving the infimum; i.e., $\min\{E(p) : E \in \mathcal{E}^c_{w,2d}(Q)\}$ is well-defined.

Note that the statement of Theorem 3 applies only to even degrees. This comes as an artifact of the proof but is in no way a severe restriction for the applications that we have in mind. The definitions of degree for SOS and PS proofs as defined in Section 2 are special cases of the definitions above for appropriate choices of w and c. Thus, Theorem 3 gives Duality Theorems for them. The role of the cut-off function c in our application below will be explained in due time; i.e., after its use in the unrestricting Lemma 6 below. It is important for the lemmas that follow that these duality theorems are tight in two ways: that they have zero duality gap *and* that they respect the degree; i.e., the degree bound is the same for proofs and pseudo-expectations. We defer the proof of Theorem 3 to Section 5 where a more general statement is proved.

3.2 Unrestricting lemmas

For this section, fix three positive integers n, d and w for the numbers of pairs of twin variables, degree, and product width. We also fix an indexed set $Q = \{q_1, \ldots, q_\ell, p_1, \ldots, p_m\}$ of polynomials on the n pairs of twin variables, and a cut-off function c for Q.

▶ **Lemma 4.** *Let p and q be polynomials of degree at most $2d$. If $p \equiv q \mod I_n$, then $E(p) = E(q)$ for any $E \in \mathcal{E}^c_{w,2d}(Q)$.*

Proof. The assumption that $p \equiv q \mod I_n$ implies that both $p - q$ and $q - p$ belong to $\mathrm{PS}^c_{w,2d}(Q)$. Hence $E(p) = E(q)$ for any $E \in \mathcal{E}^c_{w,2d}(Q)$. ◀

▶ **Lemma 5.** *Let x be one of the $2n$ variables and let m be a monomial of degree at most $2d-1$. Then $E(x) = 0$ implies $E(xm) = 0$ for any $E \in \mathcal{E}^c_{w,2d}(Q)$.*

Proof. Let m_1 and m_2 be two monomials of degree at most $d-1$ and d, respectively, such that $m = m_1 m_2$. Note first that $E((xm_1)^2) = 0$, since $x - (xm_1)^2 \equiv (x - xm_1)^2 \mod I_n$ and all degrees are at most $2d$. Hence, $0 = E(x) \geq E((xm_1)^2) \geq 0$ by Lemma 4. Let then $a = E(m_2^2)$ and note that $a \geq 0$. For every positive integer k we have

$$E(xm) \leq \frac{1}{2k}(E(2kxm_1m_2) + E((kxm_1 - m_2)^2)) = \frac{a}{2k},$$
$$E(xm) \geq \frac{1}{2k}(E(2kxm_1m_2) - E((kxm_1 + m_2)^2)) = -\frac{a}{2k},$$

where in both cases the equalities follow from $E((xm_1)^2) = 0$ and $E(m_2^2) = a$. Since $a \geq 0$ and the inequalities hold for every $k > 0$ it must be that $E(xm) = 0$ and the lemma is proved. ◀

For q a polynomial on the n pairs of twin variables, $i \in [n]$ an index, and $b \in \{0, 1\}$ a Boolean value, we denote by $q[i/b]$ the polynomial that results from assigning x_i to b and $\bar{x}_i$ to $1 - b$ in q. We extend the notation to indexed sets of such polynomials through $Q[i/b]$ to mean $\{q_j[i/b] : j \in [\ell]\} \cup \{p_j[i/b] : j \in [m]\}$. Note that $q_j[i/b]$ and $p_j[i/b]$ are polynomials on $n-1$ pairs of twin variables, and their degrees are at most those of q_j and p_j, respectively.

▶ **Lemma 6.** *Let $i \in [n]$, let Q_0 and Q_1 be the extensions of Q with the polynomials $p_{m+1} = x_i$ and $p_{m+1} = \bar{x}_i$, respectively, and let c' be the extension of c that maps $m+1$ to 1. The following hold:*

- **(i)** *The function c' is a cut-off function for both Q_0 and Q_1,*
- **(ii)** *If $Q[i/0] \vdash^c_{w,2d} -1 \geq 0$, then $Q_0 \vdash^{c'}_{w,2d} -1 \geq 0$.*
- **(iii)** *If $Q[i/1] \vdash^c_{w,2d} -1 \geq 0$, then $Q_1 \vdash^{c'}_{w,2d} -1 \geq 0$.*

Proof. (i) is obvious. By symmetry we prove only (ii). Suppose that $Q[i/0] \vdash^c_{w,2d} -1 \geq 0$, say:

$$-1 = s_0 + \sum_{J \in \mathcal{J}} s_J \prod_{j \in J} q_j[i/0] + \sum_{j \in [m]} t_j p_j[i/0] + \sum_{q \in B_n} t_q q[i/0]. \tag{5}$$

For $j \in [\ell]$, write $q_j = \sum_{\alpha \in I_j} a_{j,\alpha} x^\alpha$, let $J_j = \{\alpha \in I_j : \alpha_i \geq 1\}$ and $K_j = \{\alpha \in I_j : \alpha_i = 0 \text{ and } \alpha_{n+i} \geq 1\}$ and note that

$$q_j[i/0] = q_j + \sum_{\alpha \in J_j} a_{j,\alpha}(x^\alpha / x_i^{\alpha_i})(-x_i^{\alpha_i}) + \sum_{\alpha \in K_j} a_{j,\alpha}(x^\alpha / \bar{x}_i^{\alpha_{n+i}})(1 - \bar{x}_i^{\alpha_{n+i}}).$$

Therefore $q_j[i/0] \equiv q_j + r_j x_i \mod I_n$ where

$$r_j = \sum_{\alpha \in K_j} a_{j,\alpha}(x^\alpha / \bar{x}_i^{\alpha_{n+i}}) - \sum_{\alpha \in J_j} a_{j,\alpha}(x^\alpha / x_i^{\alpha_i}).$$

Note that $\deg(r_j) \leq \deg(q_j) - 1$ since $\alpha_i \geq 1$ for $\alpha \in J_j$ and $\alpha_{n+i} \geq 1$ for $\alpha \in K_j$. Now

$$\begin{aligned} s_J \prod_{j\in J} q_j[i/0] &\equiv s_J \prod_{j\in J}(q_j + r_j x_i) \mod I_n \\ &\equiv s_J \prod_{j\in J} q_j + \Big(\sum_{\substack{T\subseteq J \\ T\neq J}} s_J \prod_{j\in T} q_j \prod_{j\in J\setminus T} r_j\Big) x_i \mod I_n. \end{aligned}$$

Because c is a cut-off function for Q and $c'(J) = c(J)$, we have $\deg(s_J) \leq 2d - c(J) = 2d - c'(J)$. Likewise for every $T \neq J$, we have:

$$\begin{aligned} \deg\Big(s_J \prod_{j\in T} q_j \prod_{j\in J\setminus T} r_j\Big) &\leq \deg(s_J) + \sum_{j\in T} \deg(q_j) + \sum_{j\in J\setminus T} \deg(r_j) \\ &\leq 2d - c(J) + \sum_{j\in J} \deg(q_j) - 1 \leq 2d - 1 = 2d - c'(m+1). \end{aligned}$$

The second inequality follows from the facts that $J \setminus T \neq \emptyset$ and $\deg(r_j) \leq \deg(q_j) - 1$ for all $j \in [m]$, the third inequality follows from the fact that c is a cut-off function for Q, and the equality follows from the definition of c'. Hence, $Q_0 \vdash^{c'}_{w,2d} s_J \prod_{j\in J} q_j[i/0]$. A similar and easier argument with t_j and p_j in place of s_J and $\prod_{j\in J} q_j$ shows that $Q_0 \vdash^{c'}_{w,2d} t_j p_j[i/0]$. This gives proofs for all terms in the right-hand side of (5), and the proof of the lemma is complete. ◀

Some comments are in order about the role of the cut-off function in the above proof. First note that, at the semantic level, the constraint $q_j[i/0] \geq 0$ is equivalent to the pair of constraints $q_j \geq 0$ and $x_i = 0$. At the level of syntatic proofs, though, these two representations of the same constraint behave differently: although a lift $s_j q_j[i/0]$ of the restriction $q_j[i/0] \equiv q_j + r_j x_i$ of q_j may have its degree bounded by $2d$, the degree of its direct simulation through $s_j q_j + s_j r_j x_i$ could exceed $2d$. The role of the cut-off function is to restrict the lifts $s_j q_j[i/0]$ in such a way that their simulation through $s_j q_j + s_j r_j x_i$ remains a valid lift of degree at most $2d$; this is the case if, indeed, the allowed lifts $s_j q_j[i/0]$ of $q_j[i/0]$ are those satisfying $\deg(s_j) \leq 2d - c(j)$, where $c(j) \geq \deg(q_j)$. This is why c is designed to depend only on the index j (or J) and not on the polynomial indexed by j (or J).

▶ **Lemma 7.** *Let $i \in [n]$, let Q_0 and Q_1 be the extensions of Q with the polynomials $p_{m+1} = x_i$ and $p_{m+1} = \bar{x}_i$, respectively, and let c' be the extension of c that maps $m+1$ to 1. The following hold:*

- **(i)** *The function c' is a cut-off function for both Q_0 and Q_1.*
- **(ii)** *If $Q_0 \vdash^{c'}_{w,2d} -1 \geq 0$, then $E(x_i) > 0$ for any $E \in \mathcal{E}^{c}_{w,2d}(Q)$.*
- **(iii)** *If $Q_1 \vdash^{c'}_{w,2d} -1 \geq 0$, then $E(\bar{x}_i) > 0$ for any $E \in \mathcal{E}^{c}_{w,2d}(Q)$.*

Proof. (i) is obvious. We prove (ii); the proof of (iii) is symmetric. Suppose towards a contradiction that there is $E \in \mathcal{E}^{c}_{w,2d}(Q)$ such that $E(x_i) = 0$. We want to show that E is also in $\mathcal{E}^{c'}_{w,2d}(Q_0)$. This contradicts the assumption that $Q_0 \vdash^{c'}_{w,2d} -1 \geq 0$. Let

$$s_\emptyset + \sum_{J\in\mathcal{J}} s_J \prod_{j\in J} q_j + \sum_{j\in[m]} t_j p_j + t_{m+1} x_i + \sum_{q\in B_n} t_q q \tag{6}$$

be a proof from Q_0 of degree mod c' at most $2d$ and product-width at most w. First note that $\deg(t_{m+1}) \leq 2d - c'(m+1) \leq 2d - 1$. Therefore, Lemma 5 applies to all the monomials of t_{m+1}, so $E(t_{m+1} x_i) = 0$. The rest of (6) will get a non-negative value through E, since by assumption E is in $\mathcal{E}^{c}_{w,2d}(Q)$ and c is c' restricted to $\mathcal{P}([\ell]) \dot\cup [m]$. Thus, E is in $\mathcal{E}^{c'}_{w,2d}(Q_0)$. ◀

▶ **Lemma 8.** *Let $i \in [n]$ and assume that $d \geq 2$. The following hold:*
(i) *If $Q[i/0] \vdash^c_{w,2d-2} -1 \geq 0$ and $Q[i/1] \vdash^c_{2d} -1 \geq 0$, then $Q \vdash^c_{w,2d} -1 \geq 0$.*
(ii) *If $Q[i/0] \vdash^c_{w,2d} -1 \geq 0$ and $Q[i/1] \vdash^c_{2d-2} -1 \geq 0$, then $Q \vdash^c_{w,2d} -1 \geq 0$.*

Proof. Since in this proof c and w remain fixed, we write $\vdash_{2d}$ instead of $\vdash^c_{w,2d}$ and $\mathcal{E}_{2d}(Q)$ instead of $\mathcal{E}^c_{w,2d}(Q)$, and act similarly for degree $2d-2$. First note that $-\bar{x}_i x_i = (x_i^2 - x_i) - x_i(x_i + \bar{x}_i - 1)$, and $d \geq 1$, so

$$\vdash_{2d} -\bar{x}_i x_i \geq 0. \tag{7}$$

We prove (i); the proof of (ii) is entirely analogous.

Assume $Q[i/0] \vdash_{2d-2} -1 \geq 0$. By Lemmas 6 and 7 and $d \geq 2$ we have $E(x_i) > 0$ for any $E \in \mathcal{E}_{2d-2}(Q)$. Then, by the Duality Theorem, there exist $\epsilon > 0$ such that $Q \vdash_{2d-2} x_i \geq \epsilon$. To see this, let $\gamma = \sup\{r \in \mathbb{R} : Q \vdash_{2d-2} x_i \geq r\} = \inf\{E(x_i) : E \in \mathcal{E}_{2d-2}(Q)\}$. If $\mathcal{E}_{2d-2}(Q)$ is empty, then $\gamma = +\infty$ and any $\epsilon > 0$ serves the purpose. If $\mathcal{E}_{2d-2}(Q)$ is non-empty, then the Duality Theorem says that the infimum is achieved, hence $\gamma = E(x_i) > 0$ for some E in $\mathcal{E}_{2d-2}(Q)$, and $\epsilon = \gamma/2 > 0$ serves the purpose. Using $d \geq 2$ again, $Q \vdash_{2d} \bar{x}_i^2 x_i \geq \bar{x}_i^2 \epsilon$, so

$$Q \vdash_{2d} \bar{x}_i x_i \geq \bar{x}_i \epsilon. \tag{8}$$

Assume also $Q[i/1] \vdash_{2d} -1 \geq 0$. By Lemmas 6 and 7 we have $E(\bar{x}_i) > 0$ for any $E \in \mathcal{E}_{2d}(Q)$, and this time $d \geq 1$ suffices. By the same argument as before, by the Duality Theorem there exist $\delta > 0$ such that $Q \vdash_{2d} \bar{x}_i \geq \delta$. Now $d \geq 1$ suffices to get

$$Q \vdash_{2d} \bar{x}_i \epsilon \geq \delta\epsilon. \tag{9}$$

Adding (7), (8) and (9) gives $Q \vdash_{2d} 0 \geq \delta\epsilon$, i.e., $Q \vdash_{2d} -1 \geq 0$. ◀

3.3 Inductive proof

We need one more technical concept: a PS proof as in (4) is *multilinear* if s_0 and s_J are sums-of-squares of multilinear polynomials for each $J \in \mathcal{J}$, and t_j is a multilinear polynomial for each $j \in [m]$.

▶ **Lemma 9.** *For every two positive integers s and w and every indexed set Q of polynomials, if there is a PS refutation from Q of monomial size at most s and product-width at most w, then there is a multilinear PS refutation from Q of monomial size at most s and product-width at most w.*

Proof. Assume that $Q = \{q_1, \ldots, q_\ell, p_1, \ldots, p_m\}$ and that there is a refutation from Q as in (4), with $s_0 = \sum_{i=1}^{k_0} r_{i,0}^2$ and $s_J = \sum_{i=1}^{k_J} r_{i,J}^2$ for $J \in \mathcal{J}$, where the total number of monomials among the $r_{i,0}$, $r_{i,J}$ and t_j is at most s. For each polynomial r let $\overline{r}$ be its direct multilinearization; i.e., each power x^l with $l \geq 2$ that appears in r is replaced by x. It is obvious that $r \equiv \overline{r} \mod I_n$ and also $r^2 \equiv \overline{r}^2 \mod I_n$, where n is the number of pairs of twin variables in Q. Moreover, the number of monomials in $\overline{r}$ does not exceed that of r. Thus, setting $s'_0 = \sum_{i=1}^{k_0} \overline{r_{i,0}}^2$, $s'_J = \sum_{i=1}^{k_J} \overline{r_{i,J}}^2$ and $t'_j = \overline{t_j}$ we get

$$-1 \equiv s'_0 + \sum_{J \in \mathcal{J}} s'_J \prod_{j \in J} q_j + \sum_{j \in [m]} t'_j p_j \mod I_n,$$

It follows that Q has a multilinear refutation of monomial size at most s. ◀

Theorem 1 will be a consequence of the following lemma for a suitable choice of d and c:

▶ **Lemma 10.** *For every natural number n, every indexed set Q of polynomials with n pairs of twin variables, every cut-off function c for Q, every real $s \geq 1$ and every two positive integers w and d, if there is a multilinear PS refutation from Q of product-width at most w with at most s many explicit monomials of degree at least d (counted with multiplicity), then there is a PS refutation from Q of product-width at most w and degree mod c at most $2d' + 2d''$ where $d' = d + \lfloor 2(n+1)\log(s)/d \rfloor$ and $d'' = \max\{1, \lceil (\max c)/2 \rceil\}$.*

Proof. The proof is an induction on n. Let Q be an indexed set of polynomials with n pairs of twin variables, let c be a cut-off function for Q, let $s \geq 1$ be a real, let w and d be positive integers, and let Π be a multilinear refutation from Q of product-width at most w and at most s many explicit monomials of degree at least d. For $n = 0$ the statement is true because $2d'' \geq 2\lceil (\max c)/2 \rceil \geq \max c$. Assume now that $n \geq 1$. Let $t \leq s$ be the exact number of explicit monomials of degree at least d in Π. The total number of variable occurrences in such monomials is at least dt. Therefore, there exists one among the $2n$ variables that appears in at least $dt/2n$ of the explicit monomials of degree at least d. Let $i \in [n]$ be the index of such a variable, basic or twin. If it is basic, let $a = 0$. If it is twin, let $a = 1$. Our goal is to show that

$$Q[i/a] \vdash^c_{2d'+2d''-2} -1 \geq 0 \quad \text{and} \quad Q[i/1-a] \vdash^c_{2d'+2d''} -1 \geq 0, \tag{10}$$

for d' and d'' as stated in the lemma. If we achieve so, then $d' + d'' \geq 2$ because $d' \geq d \geq 1$ and $d'' \geq 1$, so Lemma 8 applies on (10) to give $Q \vdash^c_{2d'+2d''} -1 \geq 0$, which is what we are after.

Consider $Q[i/a]$ first. This is a set of polynomials on $n-1$ pairs of twin variables, and $\Pi[i/a]$ is a multilinear refutation from it of product-width at most w that has at most $s' := t(1 - d/2n)$ explicit monomials of degree at least d. Moreover c is a cut-off function for it. We distinguish the cases $s' < 1$ and $s' \geq 1$. If $s' < 1$, then all explicit monomials in $\Pi[i/a]$ have degree at most $d-1$. Since $2d'' \geq \max c$, this refutation has degree mod c at most $2(d-1) + 2d'' \leq 2d' + 2d'' - 2$. This gives the first part of (10). If $s' \geq 1$, then first note that $d < 2n$. Moreover, the induction hypothesis applied to $Q[i/a]$ and s', and the same c, d and w, gives that there is a refutation from $Q[i/a]$ of product-width at most w and degree mod c at most $2d_a + 2d''$, where

$$d_a = d + \lfloor 2n\log(t(1 - d/2n))/d \rfloor \leq d + \lfloor 2(n+1)\log(s)/d \rfloor - 1.$$

Here we used the inequality $\log(1+x) \leq x$ which holds true for every real $x > -1$, and the fact that $d < 2n$. This gives the first part of (10) since $d_a \leq d' - 1$.

Consider $Q[i/1-a]$ next. In this case, the best we can say is that c is still a cut-off function for it, and that $\Pi[i/1-a]$ is a multilinear refutation from it of product-width at most w, that still has at most s many explicit monomials of degree at least d. But $Q[i/1-a]$ has at most $n-1$ pairs of twin variables, so the induction hypothesis applies to it. Applied to the same c, s, d and w, it gives that there is a refutation from $Q[i/1-a]$ of degree mod c at most $2d_{1-a} + 2d''$, where

$$d_{1-a} = d + \lfloor 2n\log(s)/d \rfloor \leq d + \lfloor 2(n+1)\log(s)/d \rfloor.$$

This gives the second part of (10) since $d_{1-a} \leq d'$. The proof is complete. ◀

Proof of Theorem 1. Assume that Q has a refutation of product-width at most w and monomial size at most s. Applying Lemma 9 we get a multilinear refutation with at most s many explicit monomials, and hence with at most s many explicit monomials of degree at least d_0, for any d_0 of our choice. We choose

$$d_0 := \lfloor \sqrt{2(n+1)\log(s)} \rfloor + 1.$$

By assumption $s \geq 1$ and we chose d_0 in such a way that $d_0 \geq 1$. Thus, Lemma 10 applies to any cut-off function c for Q, in particular for the cut-off function that is kw everywhere. This gives a refutation of product-width at most w and degree mod c at most $2d' + kw + 2$ with

$$d' \leq d_0 + 2(n+1)\log(s)/d_0 \leq 2\sqrt{2(n+1)\log(s)} + 1.$$

Since a proof of product-width at most w and degree mod c at most $2d' + kw + 2$ is also a proof of standard degree at most $2d' + kw + 2$, the proof is complete. ◀

4 Applications

The obvious targets for applications of Theorem 1 are the examples from the literature that are known to require linear degree to refute. For some of them, such as Knapsack, the size lower bound that follows was already known. For some others, the application of Theorem 1 yields a new result.

A note is in order: all the examples below are either systems of polynomial equations, i.e., $\ell = 0$, or have a single inequality, i.e., $\ell = 1$. For such systems of constraints, PS and SOS are literally equivalent. For this reason, our size lower bounds for them are stated only for SOS (stating them for PS would be accurate, but also misleading).

4.1 Tseitin, Knapsack, and Random CSPs

The first set of examples that come to mind are the Tseitin formulas: If $G_n = (V, E)$ is an n-vertex graph from a family $\{G_n : n \in \mathbb{N}\}$ of constant degree regular expander graphs, then the formula TS_n has one Boolean variable x_e for each $e \in E$ and one parity constraint $\sum_{e:u\in e} x_e = 1 \mod 2$ for each $u \in V$. Whenever the degree d of the graphs is even, this is unsatisfiable when n is odd. In the encoding of the constraints given by the system of polynomial equations $Q = \{\prod_{e:u\in e}(1 - 2x_e) = -1 : u \in V\}$, the Tseitin formulas TS_n were shown to require degree $\Omega(n)$ to refute in PS in Corollary 1 from [14]. Since the number of variables of TS_n is $dn/2$, the constraints in Q are equations of degree d, and d is a constant, Theorem 1 gives:

▶ **Corollary 11.** *There exists $\epsilon \in \mathbb{R}_{>0}$ such that for every sufficiently large $n \in \mathbb{N}$, every SOS refutation of TS_n has monomial size at least $2^{\epsilon n}$.*

Among the semialgebraic proof systems in the literature, exponential size lower bounds for Tseitin formulas were known before for a proof system called static LS_+ in [15, 18]. Up to at most doubling the degree, this can be seen as the subsystem of SOS in which every square s_j is of the very special form

$$s_j = \Big(\Big(\sum_{i\in[n]} a_i x_i + b\Big)\prod_{i\in I} x_i \prod_{j\in J}(1 - x_j)\Big)^2.$$

A second set of examples are the Knapsack equations $2x_1 + \cdots + 2x_n = k$, which are unsatisfiable for odd integers k. We denote them $\mathrm{KS}_{n,k}$. These are known to require degree $\Omega(\min\{k, 2n - k\})$ to refute in SOS [13]. Since the number of variables is n and the degree is one, Theorem 1 gives an exponential size $2^{\Omega(n)}$ lower bound when $k = n$. For this example, an exponential size lower bound for SOS was also proved in Theorem 9.1 from [15] when $k = \Theta(n)$, so this result is not new. We state the precise relationship that the degree-reduction theorem gives in terms of n and k, which yields superpolynomial lower bounds for $k = \omega(\sqrt{n \log n})$.

▶ **Corollary 12.** *There exist $\epsilon \in \mathbb{R}_{>0}$ such that for every sufficiently large $n \in \mathbb{N}$ and $k \in [n]$, every SOS refutation of $\mathrm{KS}_{n,k}$ has monomial size at least $2^{\epsilon k^2/n}$.*

The third set of examples come from sparse random instances of constraint satisfaction problems. As far as we know, monomial size lower bounds for these examples do not follow from earlier published work without using our result, so we give the details.

When C is a clause with k literals, say $x_{i_1} \vee \cdots \vee x_{i_\ell} \vee \bar{x}_{i_{\ell+1}} \vee \cdots \vee \bar{x}_{i_k}$, we write p_C for the unique multilinear polynomial on the variables $x_{i_1}, \ldots, x_{i_k}$ of C that evaluates to the same truth-value as C over Boolean assignments; concretely $p_C = 1 - \prod_{j=1}^{\ell}(1 - x_{i_j}) \prod_{j=\ell+1}^{k} x_{i_j}$. More generally, if C denotes a constraint on k Boolean variables, we write p_C for the unique multilinear polynomial on the variables of C that represents C over Boolean assignments; i.e., such that $p_C(x) = 1$ if x satisfies C, and $p_C(x) = 0$ if x falsifies C, for any $x \in \{0,1\}^n$.

▶ **Theorem 13** (see Theorem 12 in [32])**.** *For every $\delta \in \mathbb{R}_{>0}$ there exist $c, \epsilon \in \mathbb{R}_{>0}$ such that, asymptotically almost surely as n goes to infinity, if $m = \lceil cn \rceil$ and $C_1, \ldots, C_m$ are random 3-XOR (resp. 3-SAT) constraints on $x_1, \ldots, x_n$ that are chosen uniformly and independently at random, then there is a degree-ϵn SOS pseudo-expectation for the system of polynomial equations $p_{C_1} = 1, \ldots, p_{C_m} = 1$, and at the same time every truth assignment for $x_1, \ldots, x_n$ satisfies at most a $1/2 + \delta$ fraction (resp. $7/8 + \delta$) of the constraints $C_1, \ldots, C_m$.*

It should be noted that it is not immediately obvious, from just reading the definitions, that the statement of Theorem 12 in [32] gives the pseudo-expectation as stated in Theorem 13. However, the proof of Theorem 12 in [32] is by now sufficiently well understood to know that Theorem 13 holds true as stated. One way of seeing this is by noting that the proof of Theorem 12 in [32] and the proof of the lower bound for the Tseitin formulas in Corollary 1 of [14] are essentially the same. In particular Theorem 12 in [32] holds true also for proving the existence of SOS pseudo-expectations as stated in Theorem 13.

As an immediate consequence we get:

▶ **Corollary 14.** *There exist $c, \epsilon \in \mathbb{R}_{>0}$ such that, asymptotically almost surely as n goes to infinity, if $m = \lceil cn \rceil$ and $C_1, \ldots, C_m$ are random 3-XOR (resp. 3-SAT) constraints on $x_1, \ldots, x_n$ that are chosen uniformly and independently at random, then every SOS refutation of $p_{C_1} = 1, \ldots, p_{C_m} = 1$ has monomial size at least $2^{\epsilon n}$.*

It is often stated that Theorem 13 gives optimal integrality gaps for the approximability of MAX-3-XOR and MAX-3-SAT by linear degree SOS. Corollary 14 is its analogue for subexponential size SOS. There is however a subtelty in that the validity of the integrality gap statement could depend on the encoding of the objective function. The next section is devoted to clarify this.

4.2 MAX-CSPs

An instance $\mathcal{I}$ of the Boolean MAX-CSP problem is a sequence $C_1, \ldots, C_m$ of constraints on n Boolean variables. We are asked to maximize the fraction of satisfied constraints. If p_j denotes the unique multilinear polynomial on the variables of C_j that represents C_j, then the *optimal value* for an instance $\mathcal{I}$ can be formulated as follows:

$$\mathrm{opt}(\mathcal{I}) := \max_{x \in \{0,1\}^n} \quad \tfrac{1}{m} \textstyle\sum_{j=1}^{m} p_j(x). \tag{11}$$

We could ask for the least upper bound on (11) that can be certified by an SOS proof of some given complexity c, i.e., monomial size at most s, degree at most $2d$, etc. There are at

least three formulations of this question. Using the notation $\vdash_c$ to denote SOS provability with complexity c, the three formulations are:

$$\mathrm{sos''}_c(\mathcal{I}) := \inf\{\gamma \in \mathbb{R} : \vdash_c \tfrac{1}{m}\textstyle\sum_{j=1}^m p_j(x) \leq \gamma\}, \tag{12}$$

$$\mathrm{sos'}_c(\mathcal{I}) := \inf\{\gamma \in \mathbb{R} : \ \{p_j(x) = y_j : j \in [m]\} \vdash_c \tfrac{1}{m}\textstyle\sum_{j=1}^m y_j \leq \gamma\}, \tag{13}$$

$$\mathrm{sos}_c(\mathcal{I}) := \inf\{\gamma \in \mathbb{R} : \ \{p_j(x) = y_j : j \in [m]\} \cup \{\tfrac{1}{m}\textstyle\sum_{j=1}^m y_j \geq \gamma\} \vdash_c -1 \geq 0\}. \tag{14}$$

The first formulation asks directly for the least upper bound on the objective function of (11) that can be certified in complexity c. The second formulation is similar but stronger since it allows m additional Boolean variables $y_1, \ldots, y_m$, and their twins. The third is the strongest of the three as it asks for the least value that can be proved impossible. In addition, unlike the other two, the set of hypotheses in (14) mixes equations and inequality constraints. It should be obvious that (for natural complexity measures) we have $\mathrm{sos}_c(\mathcal{I}) \leq \mathrm{sos'}_c(\mathcal{I}) \leq \mathrm{sos''}_c(\mathcal{I})$ so lower bounds on sos_c imply lower bounds for the other two.

Theorem 13 gives, by itself, optimal integrality gaps for MAX-3-XOR and MAX-3-SAT for linear degree SOS in the $\mathrm{sos''}_c$ formulation, when c denotes SOS-degree. However, the degree lower bound that follows from this formulation does not let us apply our main theorem; the statement is not about refutations, it is about proving an inequality, so Theorem 1 does not apply. In the following we argue that Theorem 13 also gives optimal integrality gaps in the $\mathrm{sos'}_c$ and sos_c formulations of the problems. Since the sos_c formulation *is* about refutations, our main theorem will apply.

We write $\alpha_c(\mathcal{I})$ for the supremum of the $\alpha \in [0,1]$ for which

$$\alpha \cdot \mathrm{sos}_c(\mathcal{I}) \leq \mathrm{opt}(\mathcal{I}) \leq \mathrm{sos}_c(\mathcal{I}) \tag{15}$$

holds. If $\mathcal{C}$ is a class of instances, then we write $\alpha_c^*(\mathcal{C}) := \inf\{\alpha_c(\mathcal{I}) : \mathcal{I} \in \mathcal{C}\}$; the *$sos_c$-approximation factor* for $\mathcal{C}$. It is our goal to show that Theorem 13 implies that, for SOS proofs of sublinear degree, the sos_c-approximation factor of MAX-3-XOR is at most $1/2$, and that of MAX-3-SAT is at most $7/8$. These are optimal. This will follow from Theorem 13 and the following general fact about pseudo-expectations that (pseudo-)satisfy all the constraints:

▶ **Lemma 15.** *Let $\mathcal{I}$ be a MAX-CSP instance with n Boolean variables and m constraints of arity at most k, represented by multilinear polynomials $p_1, \ldots, p_m$, and let $Q = \{p_j(x) = 1 : j \in [m]\}$ and $Q' = \{p_j(x) = y_j : j \in [m]\} \cup \{\frac{1}{m}\sum_{j=1}^m y_j \geq 1\}$. If there is a degree-$2dk$ SOS pseudo-expectation E for Q, then there is a degree-$2d$ SOS pseudo-expectation E' for Q'.*

Proof. Let σ be the substitution that sends y_j to $p_j(x)$ and $\bar{y}_j$ to $1 - p_j(x)$ for $j = 1, \ldots, m$. For each polynomial p on the x and y variables, define $E'(p) := E(p[\sigma])$, where $p[\sigma]$ denotes the result applying the substitution to p. The proof that this works relies on the fact that if p and q are polynomial in the x and y variables, then $(pq)[\sigma] = p[\sigma]q[\sigma]$, and $\deg((pq)[\sigma]) \leq \deg(p[\sigma]q[\sigma]) \leq 2k(\deg(p) + \deg(q))$. In particular, squares maps to squares by the substitution. It is obvious that each equation $p_j(x) = y_j$ lifts: $E'(t(p_j(x) - y_j)) = E(t[\sigma](p_j(x) - p_j(x))) = E(0) = 0$. It is equaly obvious that the inequality $\frac{1}{m}\sum_{j=1}^m y_j - 1 \geq 0$ lifts: $E'(s(\frac{1}{m}\sum_{j=1}^m y_j - 1)) = \frac{1}{m}\sum_{j=1}^m E(s[\sigma](p_j(x) - 1)) \geq 0$. This completes the proof of the lemma. ◀

Combining this with Theorem 13 and Theorem 1 we get:

▶ **Corollary 16.** *For every $\delta \in \mathbb{R}_{>0}$, there exist $r, \epsilon \in \mathbb{R}_{>0}$ such that if c denotes SOS monomial size at most $2^{\epsilon n}$, where n is the number of variables, then $\alpha_c^*(\text{MAX-3-XOR}) \leq 1/2 + \delta$ (resp. $\alpha_c^*(\text{MAX-3-SAT}) \leq 7/8 + \delta$), and the gap is witnessed by an instance $\mathcal{I}$ with $m = \lceil rn \rceil$ many uniformly and independently chosen random constraints, for which $\mathrm{sos}_c(\mathcal{I}) = 1$ and $\mathrm{opt}(\mathcal{I}) \leq 1/2 + \delta$ (resp. $\mathrm{opt}(\mathcal{I}) \leq 7/8 + \delta$), asymptotically almost surely as n goes to infinity.*

5 Duality

In this section we finally prove the stated Duality Theorem for PS in a more general setting. We start by recalling some basic facts about ordered vector spaces from [28]. We prove the results for pre-ordered vector spaces rather than ordered ones since the polynomial spaces we will apply the results to carry a natural pre-order.

5.1 Vector spaces with order unit

A pre-ordered vector space is a pair $\langle V, \leq \rangle$, where V is a real vector space and $\leq$ is a pre-order that respects vector addition and multiplication by a non-negative scalar, i.e. the following hold for all $p, q, p_1, p_2, q_1, q_2 \in V$ and $a \in \mathbb{R}_{\geq 0}$:

(i) $p_1 \leq q_1$ and $p_2 \leq q_2$ only if $p_1 + p_2 \leq q_1 + q_2$;

(ii) $p \leq q$ only if $ap \leq aq$.

Pre-ordered vector spaces arise naturally from convex cones of real vector spaces. If $C \subseteq V$ is a convex cone, then the relation defined by $p \leq_C q$ if $q - p \in C$ satisfies the above requirements. An element $e \in V$ is an *order unit* for $\langle V, \leq \rangle$ if for any $p \in V$ there is some $r \in \mathbb{R}_{\geq 0}$ such that $re \geq p$.

For the rest of this section let $\langle V, \leq \rangle$ be a pre-ordered vector space with an order unit e.

▶ **Lemma 17.** *The following hold.*

(i) $e \geq 0$*;*

(ii) *For every* $p \in V$ *and* $r_1, r_2 \in \mathbb{R}$ *with* $r_1 \leq r_2$*, if* $r_1 e \geq p$*, then* $r_2 e \geq p$*.*

(iii) *For every* $p \in V$ *there is* $r \in \mathbb{R}_{\geq 0}$ *such that* $re \geq p \geq -re$*;*

(iv) *If* $-e \geq 0$*, then* $p \geq 0$ *for every* $p \in V$*.*

Proof. (i) There is some $r \in \mathbb{R}_{\geq 0}$ such that $re \geq -e$, i.e. $(r+1)e \geq 0$, and so $e \geq 0$. (ii) Now $r_2 - r_1 \geq 0$ and so $(r_2 - r_1)e \geq 0$. Thus $(r_2 - r_1)e + r_1 e \geq p$, i.e. $r_2 e \geq p$. (iii) Let r_1 be such that $r_1 e \geq p$ and let r_2 be such that $r_2 e \geq -p$, and let $r = \max\{r_1, r_2\}$. Now $re \geq p \geq -re$. (iv) Suppose $-e \geq 0$ and let $r \in \mathbb{R}_{\geq 0}$ be such that $re \geq -p$. Now also $-re \geq 0$ and so $0 \geq -p$, i.e. $p \geq 0$. ◀

Let U be a subspace of V. A linear functional $L \colon U \to \mathbb{R}$ is *positive* if $u \geq 0$ implies $L(u) \geq 0$ for all $u \in U$. Equivalently, L is positive if it is order-preserving, i.e., if $u \leq v$ implies $L(u) \leq L(v)$ for all $u, v \in U$. A positive linear functional L on V is a *pseudo-expectation* if $L(e) = 1$. We denote the set of all pseudo-expectations of V by $\mathcal{E}(V)$.

Suppose U contains the order unit and let $p \in V$. By Lemma 17.(iii) the following two sets are non-empty:

$$p \downarrow U = \{v \in U : p \geq v\},$$
$$p \uparrow U = \{v \in U : v \geq p\}.$$

If L is any positive linear functional that is defined on U, then $d_p^L = \sup\{L(v) : v \in p \downarrow U\}$ and $u_p^L = \inf\{L(v) : v \in p \uparrow U\}$ are real numbers and $d_p^L \leq u_p^L$. Note also that if $p \in U$, then $d_p^L = L(p) = u_p^L$.

▶ **Lemma 18.** *Let* U *be a subspace of* V *containing the order unit* e*, and let* L *be a positive linear functional on* U*. Then for any* $p \in V \setminus U$ *and for any* $\gamma \in \mathbb{R}$ *satisfying* $d_p^L \leq \gamma \leq u_p^L$ *there is a positive linear functional* L' *that is defined on* $\operatorname{span}(\{p\} \cup U)$*, that extends* L*, and such that* $L'(p) = \gamma$*.*

Proof. Every element of $\text{span}(\{p\} \cup U)$ can be written uniquely in form $ap + v$, where $a \in \mathbb{R}$ and $v \in U$. Define L' by

$$L'(ap + v) = a\gamma + L(v).$$

It is easy to check that L' is linear map. We show that L' is positive by considering a few cases.

Case (i) $a = 0$. If $ap + v \geq 0$ and $a = 0$, then $v \geq 0$ and $L'(ap + v) = L(v) \geq 0$. Case (ii) $a > 0$. Suppose that $ap+v \geq 0$ and $a > 0$. Then $p \geq -(v/a)$, and so $L(-(v/a)) \leq \gamma$, i.e. $0 \leq a\gamma + L(v)$. Case (iii) $a < 0$. Suppose that $ap + v \geq 0$ and $a < 0$. Then $-a > 0$, and so $-(v/a) \geq p$. Hence $\gamma \leq L(-(v/a))$, and so $0 \leq a\gamma + L(v)$. ◀

Now we can prove the general duality theorem for pre-ordered vector spaces that admit an order unit. For a more general version of this result, see [28].

▶ **Theorem 19.** *For any $p \in V$ it holds that*

$$\sup\{r \in \mathbb{R} : p \geq re\} = \inf\{E(p) : E \in \mathcal{E}(V)\}.$$

Moreover, if the set $\mathcal{E}(V)$ is non-empty, then there is a pseudo-expectation achieving the infimum, i.e., $\min\{E(p) : E \in \mathcal{E}(V)\}$ *is well-defined.*

Proof. The inequality from left to right is clear. For the inequality from right to left we distinguish two cases: whether $-e \geq 0$ or not. If $-e \geq 0$, then $\mathcal{E}(V) = \emptyset$, since $-1 \not\geq 0$, so $\inf\{E(p) : E \in \mathcal{E}(V)\} = +\infty$. On the other hand $\sup\{r \in \mathbb{R} : p \geq re\} = +\infty$ by Lemma 17.*(iv)*, so the claim follows. If $-e \not\geq 0$, then $re \geq 0$ implies $r \geq 0$, so the map defined by $L_0(re) = r$ for all $r \in \mathbb{R}$ is a positive linear functional on $U_0 = \text{span}(\{e\})$. Note now that $d_p^{L_0} = \sup\{r \in \mathbb{R} : p \geq re\}$, and so, to prove the theorem, it suffices to show that there is some pseudo-expectation E extending L_0 such that $E(p) = d_p^{L_0}$.

If $p \in U_0$, then $L_0(p) = d_p^{L_0}$. On the other hand if $p \notin U_0$, then by Lemma 18, there is a positive linear functional L' extending L_0 on $\text{span}(\{e, p\})$ such that $L'(p) = d_p^{L_0}$. Now consider the set $\mathcal{A}$ of all positive linear functionals L that are defined on a subspace $U \subseteq V$ containing both e and p, and satisfy $L(e) = 1$ and $L(p) = d_p^{L_0}$. By the argument above $\mathcal{A} \neq \emptyset$. On the other hand $\mathcal{A}$ is closed under unions of chains and so, by Zorn's lemma, there is some maximal $E \in \mathcal{A}$.

Now the domain of E is the whole of V, since otherwise we could extend E by using Lemma 18, contradicting the maximality of E. Hence E is the pseudo-expectation we are after. ◀

5.2 Order units for semi-algebraic proof systems

For the purposes of this section we define a more general notion of Positivstellensatz proof that works modulo an arbitrary ideal I, not only the Boolean ideal I_n. Let I be an ideal of the polynomial space $\mathbb{R}[x]$, and let $Q = \{q_1 \geq 0, \ldots, q_\ell \geq 0, p_1 = 0, \ldots, p_m = 0\}$ be a set of constraints. A PS proof mod I of $p \geq 0$ from Q is an identity (of $\mathbb{R}[x]/I$) of the form

$$p \equiv s_\emptyset + \sum_{J \subseteq \mathcal{J}} s_J \prod_{j \in J} q_j + \sum_{j \in [m]} t_j p_j \mod I, \tag{16}$$

where $\mathcal{J}$ is a collection of non-empty subsets of $[\ell]$, each s_J is a sum-of-squares polynomial, $s_J = \sum_{i=1}^{k_J} r_{i,J}^2$, and each t_j is an arbitrary polynomial.

A cut-off function for Q is a function $c\colon \mathcal{P}([\ell]) \dot\cup [m] \to \mathbb{N}$ with $c(J) \geq \sum_{j\in J} \deg(q_j)$ for each $J \subseteq [\ell]$, and $c(j) \geq \deg(p_j)$ for each $j \in [m]$. A PS proof as in (16) has *degree mod* c at most d if $\deg(p) \leq d$, $\deg(s_0) \leq d$, $\deg(s_J) \leq d - c(J)$ for each $J \in \mathcal{J}$, and $\deg(t_j) \leq d - c(j)$ for each $j \in [m]$. It has *product-width* at most w if each $J \in \mathcal{J}$ has cardinality at most w. We write $\mathrm{PS}^{c,I}_{w,d}(Q)$ for the convex cone of all polynomials p such that $p \geq 0$ has a PS proof mod I from Q of degree mod c at most d and product-width at most w. We will write $Q \vdash^{c,I}_{w,d} p \geq q$ if $p - q \in \mathrm{PS}^{c,I}_{w,d}(Q)$, and denote by $\mathcal{E}^{c,I}_{w,2d}(Q)$ the set of pseudo-expectations over the pre-ordered vector space determined by this cone. These definitions agree with those used in Section 3 when $I = I_n$.

We show that over any ideal I, any cut-off function c and any product-width w, if Q proves that each variable is bounded in degree two, then the constant polynomial 1 is an order unit for Q. We prove this in a series of lemmas. In order to simplify the notation, for these lemmas we write $\vdash_d$ instead of $\vdash^{c,I}_{w,d}$.

▶ **Lemma 20.** *If $Q \vdash_2 R \geq x^2$ for every variable x for some $R \in \mathbb{R}_{\geq 0}$, then for any monomial m of degree at most d and any $a \in \mathbb{R}$ there is $b \in \mathbb{R}_{\geq 0}$ such that*

$$Q \vdash_{2d} am^2 + b \geq 0.$$

Proof. We prove the claim by induction on the degree of m. If $\deg(m) = 0$, then the claim is trivial. Suppose then that $\deg(m) > 0$. If $a \geq 0$, then the claim is again clear: $am^2 = (\sqrt{a}m)^2$. Suppose that $a < 0$ and let x and m_0 be such that $m = xm_0$. By assumption $Q \vdash_2 R - x^2 \geq 0$, and so $Q \vdash_{2d} (\sqrt{-a}m_0)^2(R - x^2) \geq 0$. By induction hypothesis applied to m_0 and aR there is $b_0 \in \mathbb{R}_{\geq 0}$ such that $Q \vdash_{2d} aRm_0^2 + b_0 \geq 0$. By adding we have that $Q \vdash_{2d} am^2 + b_0 \geq 0$. ◀

▶ **Lemma 21.** *If $Q \vdash_2 R \geq x^2$ for every variable x for some $R \in \mathbb{R}_{\geq 0}$, then for any monomial m of degree at most $2d$ and any $a \in \mathbb{R}$ there is $b \in \mathbb{R}_{\geq 0}$ such that*

$$Q \vdash_{2d} am + b \geq 0.$$

Proof. Let m_0 and m_1 be monomials of degree at most d such that $m = m_0m_1$. Now if $a \geq 0$, then $(\sqrt{a/2}m_0 + \sqrt{a/2}m_1)^2 = (a/2)m_0^2 + am + (a/2)m_1^2$. Now, by previous lemma, there are non-negative b_0 and b_1 such that $Q \vdash_{2d} (-a/2)m_i^2 + b_i \geq 0$ for $i \in \{0, 1\}$. Hence $Q \vdash_{2d} am + b_0 + b_1 \geq 0$. If $a < 0$, then $(\sqrt{-a/2}m_0 - \sqrt{-a/2}m_1)^2 = (-a/2)m_0^2 + am + (-a/2)m_1^2$. Now, again by previous lemma, there are non-negative b_0 and b_1 such that $Q \vdash_{2d} (a/2)m_i^2 + b_i \geq 0$ for $i \in \{0, 1\}$. Hence $Q \vdash_{2d} am + b_0 + b_1 \geq 0$. ◀

▶ **Lemma 22.** *If $Q \vdash_2 R \geq x^2$ for every variable x for some $R \in \mathbb{R}_{\geq 0}$, then for any polynomial p of degree at most $2d$ there is $r \in \mathbb{R}_{\geq 0}$ such that*

$$Q \vdash_{2d} r \geq p.$$

Proof. Immediate from Lemma 21. ◀

This establishes the existence of an order-unit and hence, by Theorem 19, we have:

▶ **Corollary 23.** *Let d be a positive integer, let Q be an indexed set of polynomials, let c be a cut-off function for Q, let w be a positive integer, let I be an ideal of $\mathbb{R}[x]$, and let p be a polynomial of degree at most $2d$. If $Q \vdash^{c,I}_{w,2} R \geq x^2$ for every variable x for some $R \in \mathbb{R}_{\geq 0}$, then*

$$\sup\{r \in \mathbb{R} : Q \vdash^{c,I}_{w,2d} p \geq r\} = \inf\{E(p) : E \in \mathcal{E}^{c,I}_{w,2d}(Q)\}.$$

Moreover, if the set $\mathcal{E}^{c,I}_{w,2d}(Q)$ is non-empty, then there is a pseudo-expectation achieving the infimum; i.e., $\min\{E(p) : E \in \mathcal{E}^{c,I}_{w,2d}(Q)\}$ is well-defined.

For the Boolean ideal I_n, the assumption that $Q \vdash^{c,I}_{w,2} R \geq x^2$ holds for every variable x is fulfilled with $R = 1$ since $1 - x^2 \equiv (1-x)^2 \mod I_n$. This gives Theorem 3. In the ± 1 representation of the Boolean hypercube, i.e., modulo the ideal I'_n generated by the axioms $B'_n := \{1 - x_i^2, 1 - \bar{x_i}^2, x_i + \bar{x}_i : i \in [n]\}$, the assumption is fulfilled also with $R = 1$ since in this case $1 - x^2 \equiv 0 \mod I'_n$.

6 Concluding Remarks

In this paper we addressed the question of size-degree trade-offs for PS and SOS. Some questions remain open. Most importantly, is the $O(\sqrt{n \log(s)} + kw)$ upper bound in the degree-reduction lemma tight? For Resolution and PC, whose size-width/degree trade-offs adopt the same form, the bound is known to be tight. In both cases the Ordering Principle (OP) witnesses the necessity of the square root of the number of variables in the upper bound [7, 12]. In this respect, it should be noted that it was recently shown that OP_n, which has $N = n^2$ variables, can be refuted in degree $O(\sqrt{n})$, whence degree $O(\sqrt[4]{N})$, in SOS [30]. Since the relationship between N and $\sqrt{n}$ is a 4-th root, this means that OP_n cannot be used for witnessing the necessity of the square root of the number of variables in our theorem. But can OP_n be used to show that at least some fixed root $\sqrt[r]{n}$ of n is required? So far, the best SOS degree lower bound for OP_n known is superconstant [30].

Although it looks unlikely that the dependence of $O(\sqrt{n \log(s)} + kw)$ on the product-width w could be improved by refining the current method, it is not even known whether there are examples that separate PS from SOS. Could PS collapse to SOS with respect to size or degree? Related to this, a comment worth making is that there is a general well-known technique for transforming inequalities $P \geq 0$ into equalities $P - z^2 = 0$, where z is a fresh variable. This looks relevant since, in the absence of inequalities, PS collapses to SOS just by definition. On the other hand, note that the new variable z that is introduced by this method is not Boolean, which takes us outside the Boolean hypercube.

References

1 Michael Alekhnovich, Eli Ben-Sasson, Alexander A. Razborov, and Avi Wigderson. Space Complexity in Propositional Calculus. *SIAM J. Comput.*, 31(4):1184–1211, 2002. Preliminary version in STOC 2000. doi:10.1137/S0097539700366735.

2 Albert Atserias, Massimo Lauria, and Jakob Nordström. Narrow Proofs May Be Maximally Long. *ACM Trans. Comput. Log.*, 17(3):19:1–19:30, 2016. Preliminary version in CCC 2014. doi:10.1145/2898435.

3 Boaz Barak, Fernando G. S. L. Brandão, Aram Wettroth Harrow, Jonathan A. Kelner, David Steurer, and Yuan Zhou. Hypercontractivity, sum-of-squares proofs, and their applications. In *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 307–326, 2012. doi:10.1145/2213977.2214006.

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*, s3-73(1):1–26, 1996. doi:10.1112/plms/s3-73.1.1.

5 Paul Beame and Toniann Pitassi. Simplified and Improved Resolution Lower Bounds. In *37th Annual Symposium on Foundations of Computer Science, FOCS '96, Burlington, Vermont, USA, 14-16 October, 1996*, pages 274–282, 1996. doi:10.1109/SFCS.1996.548486.

6 Eli Ben-Sasson and Avi Wigderson. Short proofs are narrow - resolution made simple. *J. ACM*, 48(2):149–169, 2001. Preliminary version in STOC 1999. doi:10.1145/375827.375835.

7 Maria Luisa Bonet and Nicola Galesi. Optimality of size-width tradeoffs for resolution. *Computational Complexity*, 10(4):261–276, 2001. doi:10.1007/s000370100000.

8 Eden Chlamtac and Madhur Tulsiani. Convex Relaxations and Integrality Gaps. In Miguel F. Anjos and Jean Bernard Lasserre, editors, *Handbook on Semidefinite, Conic and Polynomial Optimization*, pages 139–169. Springer US, Boston, MA, 2012. doi:10.1007/978-1-4614-0769-0_6.

9 Matthew Clegg, Jeff 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, Pennsylvania, USA, May 22-24, 1996*, pages 174–183, 1996. doi:10.1145/237814.237860.

10 David A Cox, John Little, and Donal O'Shea. *Ideals, Varieties, and Algorithms*. Undergraduate Texts in Mathematics. Springer International Publishing, fourth edition, 2015. doi:10.1007/978-3-319-16721-3.

11 Samuel Fiorini, Serge Massar, Sebastian Pokutta, Hans Raj Tiwary, and Ronald de Wolf. Linear vs. semidefinite extended formulations: exponential separation and strong lower bounds. In *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 95–106, 2012. doi:10.1145/2213977.2213988.

12 Nicola Galesi and Massimo Lauria. Optimality of size-degree tradeoffs for polynomial calculus. *ACM Trans. Comput. Log.*, 12(1):4:1–4:22, 2010. doi:10.1145/1838552.1838556.

13 Dima Grigoriev. Complexity of Positivstellensatz proofs for the knapsack. *Computational Complexity*, 10(2):139–154, 2001. doi:10.1007/s00037-001-8192-0.

14 Dima Grigoriev. Linear lower bound on degrees of Positivstellensatz calculus proofs for the parity. *Theor. Comput. Sci.*, 259(1-2):613–622, 2001. doi:10.1016/S0304-3975(00)00157-2.

15 Dima Grigoriev, Edward A. Hirsch, and Dmitrii V. Pasechnik. Complexity of Semi-algebraic Proofs. In *STACS 2002, 19th Annual Symposium on Theoretical Aspects of Computer Science, Antibes - Juan les Pins, France, March 14-16, 2002, Proceedings*, pages 419–430, 2002. doi:10.1007/3-540-45841-7_34.

16 Dima Grigoriev and Nicolai Vorobjov. Complexity of Null- and Positivstellensatz proofs. *Annals of Pure and Applied Logic*, 113(1):153–160, 2001. First St. Petersburg Conference on Days of Logic and Computability. doi:10.1016/S0168-0072(01)00055-0.

17 Russell Impagliazzo, Pavel Pudlák, and Jirí Sgall. Lower Bounds for the Polynomial Calculus and the Gröbner Basis Algorithm. *Computational Complexity*, 8(2):127–144, 1999. doi:10.1007/s000370050024.

18 Dmitry M. Itsykson and Arist A. Kojevnikov. Lower bounds on static Lovász-Schrijver calculus proofs for Tseitin tautologies. *Journal of Mathematical Sciences*, 145(3):4942–4952, September 2007. Preliminary version in ICALP '06. doi:10.1007/s10958-007-0329-5.

19 Cédric Josz and Didier Henrion. Strong duality in Lasserre's hierarchy for polynomial optimization. *Optimization Letters*, 10(1):3–10, 2016. doi:10.1007/s11590-015-0868-5.

20 Jean-Louis Krivine. Anneaux préordonnés. *Journal d'Analyse Mathématique*, 12(1):307–326, December 1964. doi:10.1007/BF02807438.

21 Jean B. Lasserre. Global Optimization with Polynomials and the Problem of Moments. *SIAM Journal on Optimization*, 11(3):796–817, 2001. doi:10.1137/S1052623400366802.

22 Massimo Lauria and Jakob Nordström. Tight Size-Degree Bounds for Sums-of-Squares Proofs. *Computational Complexity*, 26(4):911–948, 2017. Preliminary version in CCC 2015. doi:10.1007/s00037-017-0152-4.

23 James R. Lee, Prasad Raghavendra, and David Steurer. Lower Bounds on the Size of Semidefinite Programming Relaxations. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 567–576, 2015. doi:10.1145/2746539.2746599.

24 James R. Lee, Prasad Raghavendra, David Steurer, and Ning Tan. On the Power of Symmetric LP and SDP Relaxations. In *IEEE 29th Conference on Computational Complexity, CCC 2014, Vancouver, BC, Canada, June 11-13, 2014*, pages 13–21, 2014. doi:10.1109/CCC.2014.10.

25 László Lovász and Alexander Schrijver. Cones of Matrices and Set-Functions and 0-1 Optimization. *SIAM Journal on Optimization*, 1(2):166–190, 1991. doi:10.1137/0801013.

26 Ryan O'Donnell and Yuan Zhou. Approximability and proof complexity. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, pages 1537–1556, 2013. doi:10.1137/1.9781611973105.111.

27 Pablo A. Parrilo. *Structured Semidefinite Programs and Semialgebraic Geometry Methods in Robustness and Optimization*. PhD thesis, California Institute of Technology, 2000. URL: http://resolver.caltech.edu/CaltechETD:etd-05062004-055516.

28 Vern I. Paulsen and Mark Tomforde. Vector Spaces with an Order Unit. *Indiana University Mathematics Journal*, 58(3):1319–1359, 2009. URL: http://www.jstor.org/stable/24903253.

29 Toniann Pitassi and Nathan Segerlind. Exponential Lower Bounds and Integrality Gaps for Tree-Like Lovász-Schrijver Procedures. *SIAM J. Comput.*, 41(1):128–159, 2012. Preliminary version in SODA 2009. doi:10.1137/100816833.

30 Aaron Potechin. Sum of squares bounds for the total ordering principle. *CoRR*, abs/1812.01163, 2018. arXiv:1812.01163.

31 Mihai Putinar. Positive Polynomials on Compact Semi-algebraic Sets. *Indiana University Mathematics Journal*, 42(3):969–984, 1993. URL: http://www.jstor.org/stable/24897130.

32 Grant Schoenebeck. Linear Level Lasserre Lower Bounds for Certain k-CSPs. In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, October 25-28, 2008, Philadelphia, PA, USA*, pages 593–602, 2008. doi:10.1109/FOCS.2008.74.

33 Gilbert Stengle. A nullstellensatz and a positivstellensatz in semialgebraic geometry. *Mathematische Annalen*, 207(2):87–97, June 1974. doi:10.1007/BF01362149.

Complexity Lower Bounds for Computing the Approximately-Commuting Operator Value of Non-Local Games to High Precision

Matthew Coudron
Institute for Quantum Computing, University of Waterloo, Waterloo, Canada
mcoudron@uwaterloo.ca

William Slofstra
Institute for Quantum Computing and Department of Pure Mathematics, University of Waterloo, Waterloo, Canada
weslofst@uwaterloo.ca

Abstract

We study the problem of approximating the commuting-operator value of a two-player non-local game. It is well-known that it is NP-complete to decide whether the classical value of a non-local game is 1 or $1-\epsilon$, promised that one of the two is the case. Furthermore, as long as ϵ is small enough, this result does not depend on the gap ϵ. In contrast, a recent result of Fitzsimons, Ji, Vidick, and Yuen shows that the complexity of computing the quantum value grows without bound as the gap ϵ decreases. In this paper, we show that this also holds for the commuting-operator value of a game. Specifically, in the language of multi-prover interactive proofs, we show that the power of $\mathrm{MIP}^{co}(2,1,1,s)$ (proofs with two provers, one round, completeness probability 1, soundness probability s, and commuting-operator strategies) can increase without bound as the gap $1-s$ gets arbitrarily small.

Our results also extend naturally in two ways, to perfect zero-knowledge protocols, and to lower bounds on the complexity of computing the approximately-commuting value of a game. Thus we get lower bounds on the complexity class $\text{PZK-MIP}^{co}_{\delta}(2,1,1,s)$ of perfect zero-knowledge multi-prover proofs with approximately-commuting operator strategies, as the gap $1-s$ gets arbitrarily small. While we do not know any computable time upper bound on the class MIP^{co}, a result of the first author and Vidick shows that for $s = 1 - 1/\operatorname{poly}(f(n))$ and $\delta = 1/\operatorname{poly}(f(n))$, the class $\mathrm{MIP}^{co}_{\delta}(2,1,1,s)$, with constant communication from the provers, is contained in $\mathrm{TIME}(\exp(\operatorname{poly}(f(n))))$. We give a lower bound of $\mathrm{coNTIME}(f(n))$ (ignoring constants inside the function) for this class, which is tight up to polynomial factors assuming the exponential time hypothesis.

2012 ACM Subject Classification Theory of computation → Quantum complexity theory

Keywords and phrases Quantum complexity theory, Non-local game, Multi-prover interactive proof, Entanglement

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.25

Funding *Matthew Coudron*: Supported at the IQC by Canada's NSERC and the Canadian Institute for Advanced Research (CIFAR), and through funding provided to IQC by the Government of Canada and the Province of Ontario.
William Slofstra: Supported by NSERC DG 2018-03968.

Acknowledgements We thank Zhengfeng Ji, Alex Bredariol Grilo, Anand Natarajan, Thomas Vidick, and Henry Yuen for helpful discussions.

1 Introduction

Non-local games are a subject of converging interest for quantum information theory and computational complexity theory. A central question in both fields is the complexity of approximating the optimal winning probability of a non-local game. Quantum mechanics

allows non-local strategies in which the players share entanglement, and in quantum complexity theory we are interested in understanding the optimal winning probability over these entangled strategies. Answering this question is necessary for understanding the power of multi-prover interactive proof systems with entangled provers and a classical verifier.

For classical strategies (i.e. strategies without entanglement), it is NP-hard to decide whether a non-local game has winning probability 1. The PCP theorem implies that it is NP-hard to decide whether a non-local game has winning probability 1 or winning probability $1 - \epsilon$, where ϵ is constant, promised that one of the two is the case [1, 2]. Therefore, for classical games, the complexity of computing the winning probability is the same for constant error as for zero error.

Two models for quantum strategies have historically been used when defining the entangled value of a nonlocal game: the tensor product model and the commuting-operator model. The optimal winning probability of a non-local game over tensor product strategies is called the quantum value, and optimal winning probability over all commuting-operator strategies is called the commuting-operator value.

A number of lower bounds on approximating the quantum value of a non-local game are known. In particular, Ji has shown that it is NEXP-hard to compute the quantum value of a non-local game with inverse polynomial precision, and NEEXP-hard to compute the entangled value with inverse exponential precision [11]. Fitzsimons, Ji, Vidick, and Yuen continue this line of results by showing, roughly, that for any computable function $f(n) : \mathbb{N} \to \mathbb{N}$, it is NTIME(exp($f(n)$)) hard to compute the quantum value of a nonlocal game with $1/f(n)$ precision (here n is the input size) [7]. In particular, this implies that the quantum value of a game behaves very differently from the classical winning probability, since the complexity of computing the quantum value increases without bound as the required precision increases.

It is also natural to ask whether one might be able to approximate the commuting-operator value of a game efficiently. The study of the commuting-operator value goes back to [9], where it is shown that it is NP-hard to distinguish whether the commuting operator value is 1 or 1-1/poly(n). The complexity of the commuting operator value does not seem to be explicitly studied in more recent work.

In this paper, we look at lower bounds on the complexity of approximating the commuting-operator value of linear system nonlocal games, a type of nonlocal game closely connected with the theory of finitely-presented groups [3]. We show that group-theoretic methods can be used to lower bound the complexity of approximating the commuting-operator value of a linear system nonlocal game. In particular we show that, just as with the quantum value of a game, the complexity of computing the commuting operator value of a non-local game to precision ϵ grows arbitrarily large as ϵ decreases. Because our results are based on group-theoretic methods, we observe that they naturally extend to lower bounds on approximately-commuting-operator strategies for games, a generalization of commuting-operator strategies in which Alice and Bob's strategies can interact slightly, but in such a way that the interaction is bounded by a parameter δ. Thus we show:

▶ **Theorem 1.** *There is a universal constant k such that for every language $L \subset A^*$ over a finite alphabet A and contained in* coNTIME($f(n)$)*, where $f(n)$ is at least polynomial, there is a constant $C > 0$ and a family of two-player non-local games $(\mathcal{G}_w)_{w \in A^*}$ of size* poly($|w|$) *and computable in* poly($|w|$)*-time, such that for any $\delta = o(1/f(Cn)^{2k})$, deciding whether $\omega^{co}_\delta(\mathcal{G}_w) = 1$, or*

$$\omega^{co}_\delta(\mathcal{G}_w) \leq 1 - 1/f(Cn)^k + O(\sqrt{\delta}),$$

promised that one of the two is the case, is as hard as deciding membership of w in L.

Here $\omega_\delta^{co}(\mathcal{G}_w)$ denotes the supremum of all winning probabilities for all δ-commuting-operator strategies for the game $\mathcal{G}_w$ (see Definition 9). The proof of Theorem 1 is given in Section 5. Setting $\delta = 0$ gives a hardness result for approximating the commuting-operator value $\omega^{co} := \omega_0^{co}$ of two-player non-local games.

The proof of Theorem 1 relies on a deep group theory result of Sapir, Birget, and Rips, which shows that the acceptance problem for any Turing machine can be encoded in the word problem of a finitely-presented group, in such a way that the Dehn function of the group is equivalent to the running time of the Turing machine [16]. We then use [17] to embed this group into linear system non-local games. In the case that a word $w \in A^*$ does not belong to L, the provers demonstrate this fact by showing that a certain word in the corresponding group is not equal to the identity. In this case, the representation of the group forms the proof that the word is not equal to the identity, and this representation is used to build the provers' quantum strategy. The reason that we use commuting-operator strategies in Theorem 1, and again in Theorem 2 below, is that this representation might not be finite-dimensional.

Little is known about upper bounds on the complexity of computing the value of non-local games. Most existing proposals for an algorithm are based on a hierarchy of semi-definite programs [13, 14, 6]. It remains open whether such an algorithm can approximate the commuting-operator value of a game to any precision ϵ in finite time. However, the first author and Vidick have shown that the SDP hierarchy of [13, 14, 6] can be used to estimate (with explicit convergence bounds) the optimal value of a non-local game over approximately-commuting strategies [5]. In particular [5] gives an algorithm which, given a description of a non-local game as a truth-table of size n, can decide whether the game has commuting-operator value equal to 1, or has no δ-commuting-operator strategy with winning probability higher than $1 - \epsilon$ (for constant ϵ, promised that one of the two is the case), in time $n^{O(\mathrm{poly}(\ell,1/\delta))}$, where ℓ is the size of the output alphabet for the game. Theorem 1 shows that the dependence of this algorithm on δ is necessary. For the games $\mathcal{G}_w$ in Theorem 1, $\ell = O(1)$, and $\omega_\delta^{co}(\mathcal{G}_w) = 1$ if and only if $\omega^{co}(\mathcal{G}_w) = 1$. According to the exponential time hypothesis, we might expect that the best deterministic upper bound for coNTIME($f(n)$) is TIME($2^{\mathrm{poly}(f(n))}$). Thus, if we assume the exponential time hypothesis, the non-deterministic lower bound in Theorem 1 matches the deterministic upper bound in [5] up to polynomial factors (for families of games with a constant number of outputs).

Results about the complexity of non-local games have direct and natural implications for the power of multi-prover interactive proofs. Multi-prover interactive proofs were originally defined and studied in a purely classical setting. A seminal result of Babai, Fortnow, and Lund, which studies the class MIP of languages which admit a multi-prover interactive proof with polynomial time verifier, states that MIP = NEXP. Once again, this equality is independent of the completeness-soundness gap, as long as this gap is a large enough constant. For entangled strategies, there are, a priori, two analogs of the class MIP to consider, the class MIP* of multi-prover interactive proofs in which provers may use finite-dimensional entangled strategies, and MIP^{co}, the equivalent class with commuting-operator strategies. A result of Ito and Vidick states that the class MIP$^*(4, 1, 1, 1 - 1/\mathrm{poly}(n))$ with four provers, one round, completeness probability 1, and soundness probability 1-1/poly(n) contains NEXP [10]. Ji's result mentioned earlier for computing the quantum value of game shows that with a sufficient number of provers k, MIP$^*(k, 1, 1, 1 - 1/\exp(n))$ contains NEEXP, in contrast again to the classical case [11]. Ji's result is based on a compression theorem for non-local games, which also shows that the problem of computing the quantum value of a game is complete for MIP*.

Theorem 1 can be translated into lower bounds on $\mathrm{MIP}^{co}_{\delta}$, the class of languages with a multiprover interactive proof sound against approximately commuting strategies. Furthermore, these lower bounds also apply to the class PZK-$\mathrm{MIP}^{co}_{\delta}$ of languages which admit a perfect zero knowledge multiprover interactive proof sound against approximately commuting strategies. In a perfect zero knowledge interactive proof the provers must reveal nothing to the verifier except the proven statement itself. The formal definition of these two classes is given in Definitions 18 and 20.

▶ **Theorem 2.** *There is a universal constant k such that for any language L in* $\mathrm{NTIME}(f(n))$*, where $f(n)$ is at least polynomial, there is a constant C such that for any $\delta = o(1/f(Cn)^{2k})$,*

$$\overline{L} \in \text{PZK-}\mathrm{MIP}^{co}_{\delta}(2, 1, 1, 1 - 1/f(Cn)^k),$$

where $\overline{L}$ is the complement of L.

Note that, since the containment PZK-$\mathrm{MIP}^{co}_{\delta} \subseteq \mathrm{MIP}^{co}_{\delta}$ is immediate (see Definition 20), Theorem 2 represents both a lower bound for PZK-$\mathrm{MIP}^{co}_{\delta}$ and for $\mathrm{MIP}^{co}_{\delta}$ itself. Similarly to Theorem 1, when $\delta = 0$, we get a lower bound on the class $\mathrm{MIP}^{co} := \mathrm{MIP}^{co}_{0}$ of multi-prover interactive proofs with commuting-operator strategies, which is the direct analog of the complexity class MIP^* in the commuting operator setting (indeed, the term MIP^* has been used to denote MIP^{co} in some previous works).

One reason we are interested in the class $\mathrm{MIP}^{co}_{\delta}$ is that the algorithm of [5] mentioned above gives a (deterministic) time upper bound for $\mathrm{MIP}^{co}_{\delta}$. For protocols with constant-sized outputs, this upper bound is stated in Theorem 23. In contrast, no computable upper bounds for MIP^* or MIP^{co} are known. Combining Theorem 2 with the upper bound in of [5] gives the following series of containments (written here with constants, polynomial factors, and some parameters of the $\mathrm{MIP}^{co}_{\delta}$ notation suppressed for conciseness, including a parameter requiring a constant number of outputs). For any $\delta = o(1/f(Cn)^{2k})$:

$$\begin{aligned} \mathrm{coNTIME}(f(n)) &\subseteq \text{PZK-}\mathrm{MIP}^{co}_{\delta}(1, 1 - 1/\operatorname{poly}(f(n))) \\ &\subseteq \mathrm{MIP}^{co}_{\delta}(1, 1 - 1/\operatorname{poly}(f(n))) \\ &\subseteq \mathrm{TIME}(\exp(1/\operatorname{poly}(\delta))) \end{aligned} \tag{1.1}$$

Just as for the decision problem in Theorem 1, if we assume the exponential time hypothesis then we can consider the left hand side and right hand side of Equation 1.1 above to be matching up to polynomial factors.

Our results are complementary to the results of Fitzsimons, Ji, Vidick, and Yuen, who show qualitatively similar lower bounds for computing the quantum value of k-player games and for $\mathrm{MIP}^*(k, 1, 1, s)$, where $k \geq 15$. Their results show that MIP^* with $1/f(n)$ completeness-soundness gap contains $\mathrm{NTIME}(2^{f(n)})$, matching the pattern seen in [11] for inverse polynomial and inverse exponential gaps. In contrast, in our result the scaling of the lower bound relative to the gap is weaker, requiring gap of order $1/f(n)$ to get a lower bound of $\mathrm{coNTIME}(f(n))$, and applying to commuting-operator strategies rather than quantum strategies. However, our results apply to two-player protocols, while the results of [7] apply to protocols with 15 or more players. That we get a lower bound of $\mathrm{coNTIME}(f(n))$ rather than $\mathrm{NTIME}(2^{f(n)})$ can be explained by the fact that our lower bound extends to $\mathrm{MIP}^{co}_{\delta}$, which, with the restriction to protocols with constant-sized outputs, is contained in $\mathrm{TIME}(2^{f(n)})$. Thus our results highlight the importance of considering soundness to approximately-commuting strategies when seeking lower bounds on MIP^* and MIP^{co}. It seems to be an interesting open problem to determine whether the improved bounds of [7] can be done with algebraic methods.

2 Group theory preliminaries

Recall that a *finitely-presented group* is a group G with a fixed presentation $G = \langle S : R \rangle$, meaning that G is the quotient of the free group $\mathcal{F}(S)$ generated by a finite set S, by the normal subgroup generated by a finite set of relations $R \subseteq \mathcal{F}(S)$. If $G = \langle S : R \rangle$, and $R' \subseteq \mathcal{F}(S \cup S')$, then the notation $\langle G, S' : R' \rangle$ refers to the presentation $\langle S \cup S' : R \cup R' \rangle$. A *(group) word of length k* over the generators S is a string $s_1^{a_1} \cdots s_k^{a_k}$ where $s_i \in S$ and $a_i \in \{\pm 1\}$ for all $1 \leq i \leq k$. Such a word is said to be *reduced* if $s_i = s_{i+1}$ implies that $a_i = a_{i+1}$ for all $1 \leq i \leq k-1$. Every element $w \in \mathcal{F}(S)$ is represented by a unique reduced word, and the *length* $|w|$ *of* w is defined to be the length of this reduced word. The *word problem* for G is the problem of deciding whether the image of a given element $w \in \mathcal{F}(S)$ is equal to the identity in G, or in other words, whether the word is in the normal subgroup of $\mathcal{F}(S)$ generated by R. Since the reduced form of any non-reduced word over S can be found in time linear in the length of that non-reduced word, we can ask that inputs to the word problem be represented either as reduced or non-reduced words without changing the problem.

A *(unitary) representation* of a group G is a homomorphism $\phi : G \to \mathcal{U}(\mathcal{H})$, where $\mathcal{U}(\mathcal{H})$ is the unitary group of a Hilbert space $\mathcal{H}$. If $G = \langle S : R \rangle$ is a finitely-presented group, then a representation $\phi : G \to \mathcal{U}(\mathcal{H})$ can be specified by giving a homomorphism $\widetilde{\phi} : \mathcal{F}(S) \to \mathcal{U}(\mathcal{H})$ such that $\widetilde{\phi}(r) = 1$ for every $r \in R$. If G is a group, then $\ell^2 G$ is the Hilbert space with Hilbert basis $\mathcal{B} = \{|g\rangle : g \in G\}$. This means that every element of $\mathcal{H}$ is of the form $\sum_{g \in G} c_g |g\rangle$, where $\sum_{g \in G} |c_g|^2 \leq +\infty$. Since every group G acts on itself by both left and right multiplication, G also acts by left and right multiplication on $\mathcal{B}$. Thus G acts unitarily on $\ell^2 G$ by left and right multiplication. The resulting representations $L, R : G \to \mathcal{U}(\ell^2 G)$ are called the *left* and *right regular representations* of G, respectively.

If $w \in \mathcal{F}(S)$ is a word which is equal to the identity in G, we let $\mathrm{Area}_G(w)$ be the minimum $t \geq 1$ such that

$$w = z_1 r_1^{a_1} z_1^{-1} \cdots z_t r_t^{a_t} z_t^{-1}$$

for some $r_1, \ldots, r_t \in R$, $z_1, \ldots, z_t \in \mathcal{F}(S)$, and $a_1, \ldots, a_t \in \{\pm 1\}$.[1] The *Dehn function* Dehn_G of G is the function $\mathbb{N} \to \mathbb{N}$ defined by

$$\mathrm{Dehn}_G(n) = \max\{\mathrm{Area}_G(w) : w \in \mathcal{F}(S) \text{ has } |w| \leq n \text{ and } w = 1 \text{ in G}\}.$$

If the word problem of G is decidable, then Dehn_G is computable. Conversely, the word problem of G belongs to $\mathrm{NTIME}(\mathrm{Dehn}_G(n))$ [16]. An easy way to see that the complexity of the word problem is bounded by the Dehn function (albeit with the slightly worse upper bound of $\mathrm{NTIME}(\mathrm{poly}(\mathrm{Dehn}_G(n)))$) is through the following lemma:

▶ **Lemma 3** ([8], Lemma 2.2). *Let $G = \langle S : R \rangle$ be a finitely-presented group, and let ℓ be the length of the longest relation in R. If $w \in \mathcal{F}(S)$ is equal to the identity in G and $k = \mathrm{Area}_G(w)$, then*

$$w = z_1 r_1^{a_1} z_1^{-1} \cdots z_k r_k^{a_k} z_k^{-1}$$

where $r_1, \ldots, r_k \in R$, $z_1, \ldots, z_k \in \mathcal{F}(S)$, $a_1, \ldots, a_k \in \{\pm 1\}$, and $|z_i| \leq k\ell + \ell + |w|$ for all $1 \leq i \leq k$.

[1] $\mathrm{Area}_G(w)$ can also be defined as the minimum number of regions in a van Kampen diagram with boundary word w, and this is where the name comes from.

In general, the Dehn function can be much larger than the time-complexity of the word problem of G. However, Sapir, Birget, and Rips have shown that every recursive language can be reduced to the word problem of a finitely-presented group for which the Dehn function is polynomially equivalent to the time-complexity of the word problem. For the statement of the theorem, recall that two functions $T, T' : \mathbb{N} \to \mathbb{N}$ are said to be *(asymptotically) equivalent* if there are constants C, C' such that $T(n) \leq CT'(Cn) + Cn + C$ and $T'(n) \leq C'T(C'n) + C'n + C'$ for all $n \geq 1$.

▶ **Theorem 4** ([16], Theorem 1.3)**.** *Let A be a finite alphabet, and $L \subset A^*$ a language over A contained in* NTIME$(T(n))$*, where $T(n)$ is computable and $T(n)^4$ is at least superadditive (i.e. $T(n+m)^4 \geq T(n)^4 + T(m)^4$. Then there exists a finitely-presented group $G = \langle S : R\rangle$ and an injective function $\kappa : A^* \to \mathcal{F}(S)$, such that*
(a) *$|\kappa(u)| = O(|u|)$ and $\kappa(u)$ is computable in time $O(|u|)$,*
(b) *$u \in L$ if and only if $\kappa(u) = 1$ in G, and*
(c) *$\mathrm{Dehn}_G(n)$ is bounded by a function equivalent to $T(n)^4$.*

A group over $\mathbb{Z}_2$ is a pair (G, J) where J is a central involution, i.e. an element of the center of G with $J^2 = 1$. Usually we just write G for the pair, and refer to $J = J_G$ in the same way we refer to the identity $1 = 1_G$ of a group. When $J_G \neq 1_G$, it can be used as a substitute for -1. Theorem 4 implies that any recursive decision problem can be encoded in the word problem of a group. We want an embedding of this type where the word w is a central involution. For this, we use the following trick:

▶ **Definition 5.** *Let $G = \langle S : R\rangle$ be a finitely-presented group, and let x, J, t be indeterminates not in S. Given $w \in \mathcal{F}(S)$, let*

$$\widetilde{G}_w := \langle G, x, J, t \ : \ J^2 = 1, [g, J] = 1 \text{ for all } g \in G, \\ [x, J] = 1, [t, J] = 1, [t, [x, w]] = J\rangle,$$

where $[a, b] := aba^{-1}b^{-1}$ is the group commutator.

Note that if G is finitely-presented, then we only need to include the relations $[g, J] = 1$ for g in a generating set of G, and this gives a finite presentation of $\widetilde{G}_w$.

▶ **Lemma 6.** *Given a group $G = \langle S : R\rangle$ and a word $w \in \mathcal{F}(S)$, let $\widetilde{G}_w$ be the group defined in Definition 5. Then*
(a) *J is a central involution in $\widetilde{G}_w$,*
(b) *$w = 1$ in G if and only if $J = 1$ in $\widetilde{G}_w$, and*
(c) *if $w = 1$ in G then $\mathrm{Area}_{\widetilde{G}_w}(J) \leq 4\,\mathrm{Area}_G(w) + 1$.*

Proof. Part (a) is clear. For part (b), let

$$G' := \langle G, x, J : J^2 = [x, J] = [g, J] = 1 \text{ for all } g \in G\rangle^2,$$

The element $y = [x, w]$ is equal to 1 in G' if and only if $w = 1$. If $w \neq 1$ then y has infinite order. Hence the subgroup $\langle y, J\rangle$ is equal to $\mathbb{Z} \times \mathbb{Z}_2$ if $w \neq 1$, and $\mathbb{Z}_2$ if $w = 1$. In both cases, the homomorphism induced by $y \mapsto Jy$ and $J \mapsto J$ is an automorphism of this subgroup, and

$$\widetilde{G}_w = \langle G', t : tyt^{-1} = Jy, tJt^{-1} = J\rangle$$

is the Higman-Neumann-Neumman (HNN) extension of G' by this automorphism (we refer to [12, Chapter IV] for the properties of HNN extensions). As a result, G' is a subgroup of $\widetilde{G}_w$, and part (b) follows.

For part (c), if $w = 1$ in G, then $\mathrm{Area}_G(w^{-1}) = \mathrm{Area}_G(w)$, so

$$\mathrm{Area}_{G'}([x, w]) \leq 2\,\mathrm{Area}_G(w)$$

and similarly

$$\mathrm{Area}_{\widetilde{G}_w}([t, [x, w]]) \leq 2\,\mathrm{Area}_{G'}([x, w]) \leq 4\,\mathrm{Area}_G(w).$$

Thus we can use the relation $J = [t, [x, w]]$ to conclude that $\mathrm{Area}_{\widetilde{G}_w}(J) \leq 4\,\mathrm{Area}_G(w)+1$. ◀

The last result we include in this section is a lemma which will be used to translate area calculations into bounds on distances between vectors in Hilbert spaces. If u and v are two vectors in a Hilbert space $\mathcal{H}$, we write $u \approx_\epsilon v$ to mean that $\|u - v\| \leq \epsilon$. We use the standard terminology and notation of quantum information, so for instance, a state in a Hilbert space $\mathcal{H}$ is a unit vector $|\psi\rangle$ in $\mathcal{H}$.

▶ **Definition 7.** *Let $G = \langle S : R\rangle$ be a finitely-presented group. A (δ, ϵ)-*bipartite representation of G with respect to a state $|\psi\rangle$ in a Hilbert space $\mathcal{H}$ *is a pair of homomorphisms $\Phi, \Phi' : \mathcal{F}(S) \to \mathcal{U}(\mathcal{H})$ such that*

(i) $\Phi(r)\,|\psi\rangle \approx_\epsilon |\psi\rangle$ *for all $r \in R$,*
(ii) $\Phi(s)^{-1}\,|\psi\rangle \approx_\epsilon \Phi'(s)\,|\psi\rangle$ *for all $s \in S$, and*
(iii) $\|[\Phi(s), \Phi'(t)] - \mathbb{1}\| \leq \delta$ *for all $s, t \in S$ (here $\mathbb{1}$ represents the identity operator in $\mathcal{U}(\mathcal{H})$).*

In Part (iii) and throughout this paper the notation $\|A\|$ for an operator A refers to the operator norm of A. Part (i) of Definition 7 essentially says that Φ is an approximate representation of G with respect to the state $|\psi\rangle$. Parts (ii) and (iii) are less straightforward, but these conditions arise naturally in the theory of non-local games.

▶ **Lemma 8.** *Let (Φ, Φ') be a (δ, ϵ)-bipartite representation of a finitely-presented group $G = \langle S : R\rangle$ with respect to a state $|\psi\rangle \in \mathcal{H}$, and let ℓ be the length of the longest relation in R. If $w \in \mathcal{F}(S)$ is equal to the identity in G, then*

$$\Phi(w)\,|\psi\rangle \approx_{A(w)\cdot(\epsilon+\delta)} |\psi\rangle\,,$$

where $A(w) \leq 5\ell^2\,\mathrm{Area}_G(w)^2 + 2\ell|w|\,\mathrm{Area}_G(w)$.

Proof. If $r \in R$, then $\Phi(r)\,|\psi\rangle \approx_\epsilon |\psi\rangle$, and consequently $\Phi(r)^{-1}\,|\psi\rangle \approx_\epsilon |\psi\rangle$. Thus for any $r \in R$, $z \in \mathcal{F}(S)$, and $a \in \{\pm 1\}$,

$$\begin{aligned}\Phi(zr^a z^{-1})\,|\psi\rangle &= \Phi(z)\Phi(r)^a\Phi(z)^{-1}\,|\psi\rangle \approx_{|z|\epsilon} \Phi(z)\Phi(r)^a\Phi'(z)^{-1}\,|\psi\rangle \\ &\approx_{|r||z|\delta} \Phi(z)\Phi'(z)^{-1}\Phi(r)^a\,|\psi\rangle \approx_\epsilon \Phi(z)\Phi'(z)^{-1}\,|\psi\rangle \\ &\approx_{|z|\epsilon} \Phi(z)\Phi(z)^{-1}\,|\psi\rangle = |\psi\rangle\,.\end{aligned}$$

We conclude that $\Phi(zr^a z^{-1})\,|\psi\rangle \approx_{(2|z|+1)\epsilon+\ell|z|\delta} |\psi\rangle$. The result follows from Lemma 3. ◀

3 Approximately-commuting operator strategies and linear system games

A *two-party Bell scenario* $(\mathcal{I}_A, \mathcal{I}_B, \mathcal{O}_A^*, \mathcal{O}_B^*)$ consists of finite input sets $\mathcal{I}_A, \mathcal{I}_B$, a finite set of outputs $\mathcal{O}_A^x$ for every $x \in \mathcal{I}_A$, and a finite set of outputs $\mathcal{O}_B^y$ for every $y \in \mathcal{I}_B$.[3] The

[3] The sets $\mathcal{O}_A^x$ and $\mathcal{O}_B^y$ are often assumed to be independent of the inputs x and y. However, this assumption is not essential, since we can make the output sets independent of the input sets by adding filler answers to make all output sets the same size, and stipulating that Alice and Bob lose if they output one of the filler answers. When working with linear system games, it is more convenient to have the output sets depend on the inputs.

number of outputs in a Bell scenario is the maximum of $|\mathcal{O}_A^x|$ and $|\mathcal{O}_B^y|$ over $x \in \mathcal{I}_A$ and $y \in \mathcal{I}_B$. A *two-player non-local game* consists of a Bell scenario $(\mathcal{I}_A, \mathcal{I}_B, \mathcal{O}_A^*, \mathcal{O}_B^*)$, a function $V(\cdot,\cdot|x,y) : \mathcal{O}_A^x \times \mathcal{O}_B^y \to \{0,1\}$ for every $x \in \mathcal{I}_A$ and $y \in \mathcal{I}_B$, and a probability distribution π on $\mathcal{I}_A \times \mathcal{I}_B$. In the operational interpretation of the game, the referee sends players Alice and Bob inputs $x \in \mathcal{I}_A$ and $y \in \mathcal{I}_B$ with probability $\pi(x,y)$, the players reply with outputs $a \in \mathcal{O}_A^x$ and $b \in \mathcal{O}_B^y$, and the players win if and only if $V(a,b|x,y) = 1$.

In a non-local game, the players are not usually allowed to communicate while the game is in progress. Thus, in a quantum strategy for a game, it's assumed that each player determines their output by measuring their own local system. Locality can be enforced in one of two ways: by requiring that the joint system is the tensor product of the subsystems, or by requiring that measurement operators for different players commute with each other. Strategies of the former type are called tensor-product strategies, while strategies of the latter type are called commuting-operator strategies. Tensor-product strategies are commuting-operator strategies by definition, and finite-dimensional commuting-operator strategies can be turned into equivalent tensor-product strategies. In infinite dimensional Hilbert spaces, there are commuting-operator strategies for which the corresponding correlations do not have a tensor-product model [17]. However, it's still an open question as to whether all correlations arising from commuting-operator strategies can be realized as a limit of tensor-product strategies. By a theorem of Ozawa, this question is equivalent to the Connes embedding problem. In [15, 5], the notion of a quantum strategy has been generalized to approximately-commuting strategies, where Alice and Bob's systems are allowed to interact slightly. In this paper, we focus on the case of approximately-commuting operator strategies. Unlike [5], we use projective measurements rather than the more general POVM measurements in this definition. We refer to Remark 19 for some of the consequences of this difference.

▶ **Definition 9.** *A* δ-approximately-commuting operator strategy $\mathcal{S}$ *(or* δ-AC operator strategy *for short) for a Bell scenario* $(\mathcal{I}_A, \mathcal{I}_B, \mathcal{O}_A^*, \mathcal{O}_B^*)$ *consists of a Hilbert space* $\mathcal{H}$*, a projective measurement* $\{P_a^x\}_{a \in \mathcal{O}_A^x}$ *on* $\mathcal{H}$ *for every* $x \in \mathcal{I}_A$*, a projective measurement* $\{Q_b^y\}_{b \in \mathcal{O}_B^y}$ *on* $\mathcal{H}$ *for every* $y \in \mathcal{I}_B$*, and a state* $|\psi\rangle \in \mathcal{H}$ *such that*

$$\|P_a^x Q_b^y - Q_b^y P_a^x\| \leq \delta$$

for all $(x,y) \in \mathcal{I}_A \times \mathcal{I}_B$ *and* $(a,b) \in \mathcal{O}_A^x \times \mathcal{O}_B^y$*. A* δ-approximately-commuting quantum *(or* δ-AC quantum*)* strategy *is a* δ*-AC operator strategy in which* $\mathcal{H}$ *is finite-dimensional.*

Let $\mathcal{G} = (\mathcal{I}_A, \mathcal{I}_B, \mathcal{O}_A^*, \mathcal{O}_B^*, V, \pi)$ *be a non-local game. The* winning probability of $\mathcal{G}$ with strategy $\mathcal{S}$ *is*

$$\omega(\mathcal{G};\mathcal{S}) = \left| \sum_{x \in \mathcal{I}_A, y \in \mathcal{I}_B} \pi(x,y) \sum_{a \in \mathcal{O}_A, b \in \mathcal{O}_B} V(a,b|x,y) \langle\psi| P_a^x Q_b^y |\psi\rangle \right|.$$

The δ*-AC operator value* $\omega_\delta^{co}(\mathcal{G})$ *(resp.* δ*-AC quantum value* $\omega_\delta^*(\mathcal{G})$*) of* $\mathcal{G}$ *is defined to be the supremum of* $\omega(\mathcal{G};\mathcal{S})$ *across* δ*-AC operator strategies (resp.* δ*-AC quantum strategies).*

With this definition, a *commuting-operator strategy* is simply a 0-AC operator strategy, and the usual *commuting-operator value* of a game is $\omega^{co}(\mathcal{G}) := \omega_0^{co}(\mathcal{G})$. Since commuting-operator strategies are the same as tensor product strategies in finite dimensions, a *(tensor-product) quantum strategy* is simply a 0-AC quantum strategy, and the usual *quantum value* of a game is $\omega^*(\mathcal{G}) := \omega_0^*(\mathcal{G})$. Note that when $\delta = 0$, the absolute value can be dropped in the definition of $\omega(\mathcal{G};\mathcal{S})$. When $\delta > 0$, the values $\langle\psi| P_a^x Q_b^y |\psi\rangle$ can be complex, and the absolute value is necessary. This also means that $\omega(\mathcal{G},\mathcal{S})$ cannot necessarily be interpreted as a probability when $\mathcal{S}$ is approximately but not exactly commuting.

We look at a specific class of non-local games called linear system games. Let $Mx = c$ be an $m \times n$ linear system over $\mathbb{Z}_2$, so $M \in \mathbb{Z}_2^{m\times n}$ and $c \in \mathbb{Z}_2^m$. For each $1 \leq i \leq m$, let $V_i = \{1 \leq j \leq n : M_{ij} \neq 0\}$. The linear system game $\mathcal{G}_{Mx=c}$ is the non-local game with

$$\mathcal{I}_A = \{1, \ldots, m\}, \quad \mathcal{I}_B = \{1, \ldots, n\},$$

$$\mathcal{O}_A^i = \left\{ a \in \mathbb{Z}_2^{V_i} : \sum_{j\in V_i} a_j = c_i \right\}, \quad \mathcal{O}_B^j = \mathbb{Z}_2,$$

$$V(a, b|i, j) = \begin{cases} 1 & j \notin V_i \text{ or } a_j = b \\ 0 & \text{otherwise} \end{cases},$$

and π the uniform distribution over pairs (i, j) such that $j \in V_i$. In other words, Alice receives the index i of an equation and Bob receives the index j of a variable, chosen uniformly at random from pairs (i, j) with $j \in V_i$. Alice replies with a satisfying assignment to the variables which appear in the ith equation, and Bob replies with an assignment for the jth variable. The players win if Alice and Bob both give the same assignment to variable j.

For linear system games, it is often convenient to express strategies in terms of observables, rather than measurement operators (see, for instance, [4, 3]). If $\mathcal{S} = (\mathcal{H}, \{P_a^i\}_{a\in\mathcal{O}_A^i}, \{Q_b^j\}_{b\in\mathbb{Z}_2}, |\psi\rangle)$ is a δ-AC strategy for $\mathcal{G}_{Mx=c}$, the corresponding observables are

$$A_{ij} := \sum_{a\in\mathcal{O}_A^i} (-1)^{a_j} P_a^i \text{ for } 1 \leq i \leq m, j \in V_i, \tag{3.1}$$

and

$$B_j := Q_0^j - Q_1^j \text{ for } 1 \leq j \leq n. \tag{3.2}$$

These operators are ± 1-valued observables (meaning, self-adjoint unitary operators) satisfying the equations

$$\prod_j A_{ij}^{M_{ij}} = (-\mathbb{1})^{c_i} \text{ for all } 1 \leq i \leq m, \tag{3.3}$$

$$[A_{ij}, A_{ij'}] = \mathbb{1} \text{ whenever } j, j' \in V_i \text{ for some } 1 \leq i \leq m, \text{ and} \tag{3.4}$$

$$\|[A_{ij}, B_k] - \mathbb{1}\| \leq 2^{|V_i|+1}\delta \text{ for all } 1 \leq i \leq m, j \in V_i, \text{ and } 1 \leq k \leq n. \tag{3.5}$$

We can recover the projections P_a^i, $a \in \mathcal{O}_A^i$, and Q_b^j, $b \in \mathcal{O}_B^j$, from the observables A_{ik} and B_j via the formulas

$$P_a^i = \prod_{k\in V_i} \left(\frac{\mathbb{1} + (-1)^{a_k} A_{ik}}{2} \right) \text{ and } Q_b^j = \frac{1 + (-1)^b B_j}{2}. \tag{3.6}$$

We define *bias of strategy* $\mathcal{S}$ to be

$$\beta(\mathcal{G}_{Mx=c}; \mathcal{S}) := \sum_{1\leq i\leq m} \sum_{j\in V_i} \pi(i, j) \langle\psi| A_{ij} B_j |\psi\rangle .$$

It is not hard to see that

$$\omega(\mathcal{G}_{Mx=c};\mathcal{S}) = \frac{1}{2}|\beta(\mathcal{G}_{Mx=c},\mathcal{S})+1|,$$

so we can work with the winning probability using observables as well.

It follows from [3] that when $\delta = 0$, perfect commuting-operator strategies of $\mathcal{G}_{Mx=c}$ can be understood using the following group.

▶ **Definition 10.** *Let $Mx = c$ be an $m \times n$ linear system over $\mathbb{Z}_2$. Then the* solution group *of the system is the finitely presented group $\Gamma_{Mx=c}$ generated by $x_1, \ldots, x_n, J$, and satisfying relations*

1. $[x_i, J] = x_i^2 = J^2 = 1$ *for all* $1 \le i \le n$,
2. $\prod_j x_j^{M_{ij}} = J^{c_i}$ *for all* $1 \le j \le m$, *and*
3. $[x_j, x_k] = 1$ *if there is some* $1 \le i \le m$ *with* $M_{ij}, M_{ik} \neq 0$.

We consider $\Gamma = \Gamma_{Mx=c}$ to be a group over $\mathbb{Z}_2$ with J_Γ equal to the generator J.

In particular, we can characterize when the optimal winning probability of the game is equal to 1 using this group.

▶ **Theorem 11** ([3, 18])**.** *Let $Mx = c$ be a linear system over $\mathbb{Z}_2$. Then*

(a) $\omega^{co}(\mathcal{G}_{Mx=c}) = 1$ *if and only if* $J \neq 1$ *in* $\Gamma_{Mx=c}$, *and*

(b) $\omega^*(\mathcal{G}_{Mx=c}) = 1$ *if and only if J is non-trivial in approximate representations of* $\Gamma_{Mx=c}$.

For the definition of *non-trivial in approximate representations*, we refer to [18].

Near-perfect finite-dimensional strategies of $\mathcal{G}_{Mx=c}$ correspond to approximate representations of $\Gamma_{Mx=c}$ [18]. We want to develop this theory when $\delta > 0$.

▶ **Proposition 12.** *Let $Mx = c$ be an $m \times n$ linear system, let $V_i := \{1 \le j \le n : M_{ij} \neq 0\}$, let $r := \max_i |V_i|$ be the maximum number of non-zero entries in any row, and let $K := \sum_{i=1}^m |V_i|$ be the number of non-zero entries in M. Suppose $\mathcal{S} = (\mathcal{H}, \{P_a^x\}, \{Q_b^y\}, |\psi\rangle)$ is a δ-AC operator strategy with $\omega(\mathcal{G}_{Mx=c};\mathcal{S}) \ge 1-\epsilon$ for some $\epsilon, \delta \ge 0$. Let A_{ij}, B_k be the corresponding observables defined in Equations* (3.1) *and* (3.2)*. Then*

(a) $A_{ij}|\psi\rangle \approx_{2\sqrt{K(\epsilon+2^{r-1}\delta)}} B_j|\psi\rangle$ *for all* $1 \le i \le m$ *and* $j \in V_i$,

(b) $\prod_{j=1}^m B_{ij}^{M_{ij}}|\psi\rangle \approx_{2r\sqrt{K(\epsilon+2^{r-1}\delta)}+\binom{r}{2}2^{r+1}\delta} (-\mathbb{1})^{c_i}|\psi\rangle$ *for all* $1 \le i \le m$, *and*

(c) $[B_j, B_k]|\psi\rangle \approx_{8\sqrt{K(\epsilon+2^{r-1}\delta)}+6\cdot 2^{r+1}\delta} |\psi\rangle$ *whenever there is* $1 \le i \le m$ *with* $j, k \in V_i$.

Proof. For part (a), any two unit vectors $|\psi\rangle$ and $|\phi\rangle$ satisfy $|\psi\rangle \approx_2 |\phi\rangle$, so we can assume that $\epsilon + 2^{r-1}\delta \le 1$. Write β for $\beta(\mathcal{G}_{Mx=c},\mathcal{S})$, and observe that

$$\begin{aligned}|2\operatorname{Im}\beta| = |\beta - \overline{\beta}| &= \left|\sum_{i,j}\pi(i,j)\langle\psi|A_{ij}B_j - B_jA_{ij}|\psi\rangle\right| \\ &\le \sum_{i,j}\pi(i,j)\|A_{ij}B_j - B_jA_{ij}\| \le 2^{r+1}\delta\end{aligned}$$

by Equation (3.5). Since $\omega(\mathcal{G}_{Mx=c};\mathcal{S}) \ge 1-\epsilon$, we have that

$$(1-\epsilon)^2 \le \left|\frac{\beta+1}{2}\right|^2 = \frac{(\operatorname{Re}\beta+1)^2 + (\operatorname{Im}\beta)^2}{4} \le \frac{(\operatorname{Re}\beta+1)^2 + (2^r\delta)^2}{4}.$$

Since $A_{ij}|\psi\rangle$ and $B_j|\psi\rangle$ are unit vectors, $-1 \le \operatorname{Re}\beta \le 1$, and in particular $\operatorname{Re}\beta + 1 \ge 0$. Thus

$$\operatorname{Re}\beta + 1 \ge \sqrt{4(1-\epsilon)^2 - (2^r\delta)^2} = \sqrt{(2-2\epsilon-2^r\delta)(2-2\epsilon+2^r\delta)} \ge 2 - 2\epsilon - 2^r\delta,$$

where the last inequality holds because of the assumption $2\epsilon + 2^r\delta \leq 2$. We conclude that $\operatorname{Re}\beta \geq 1 - 2\epsilon - 2^r\delta$, or $1 - \operatorname{Re}\beta \leq 2\epsilon + 2^r\delta$.

Now $\pi(i,j) = 1/K$ for all $1 \leq i \leq m$, $j \in V_i$, so

$$1 - \operatorname{Re}\beta = \frac{1}{K}\sum_{i,j}(1 - \operatorname{Re}\langle\psi| A_{ij}B_j |\psi\rangle) \leq 2\epsilon + 2^r\delta.$$

Since $\operatorname{Re}\langle\psi| A_{ij}B_j |\psi\rangle \leq 1$, we have that $1 - \operatorname{Re}\langle\psi| A_{ij}B_j |\psi\rangle \leq 2K(\epsilon + 2^{r-1}\delta)$ for all $1 \leq i \leq m$ and $j \in V_i$. So

$$\|A_{ij}|\psi\rangle - B_j|\psi\rangle\|^2 = 2 - 2\operatorname{Re}\langle\psi| A_{ij}B_j |\psi\rangle \leq 4K(\epsilon + 2^{r-1}\delta),$$

finishing the proof of part (a).

For parts (b) and (c), let $\tau = 2\sqrt{K(\epsilon + 2^{r-1}\delta)}$. Given $1 \leq i \leq m$, let $V_i = \{j_1, \ldots, j_k\}$, where $1 \leq j_1 < \ldots < j_k \leq n$. Then

$$B_{j_1}\cdots B_{j_k}|\psi\rangle \approx_\tau B_{j_1}\cdots B_{j_{k-1}}A_{ij_k}|\psi\rangle \approx_{(k-1)2^{r+1}\delta} A_{ij_k}B_{j_1}\cdots B_{j_{k-1}}|\psi\rangle.$$

Continuing this pattern, we see that

$$B_{j_1}\cdots B_{j_k} \approx_{k\tau + \binom{k}{2}2^{r+1}\delta} A_{ij_k}A_{ij_{k-1}}\cdots A_{ij_1}|\psi\rangle = (-\mathbb{1})^{c_i}|\psi\rangle,$$

where the last equality is Equation (3.3). Part (c) follows similarly from Equation (3.4). ◀

▶ **Corollary 13.** *Using the notation and hypotheses of Proposition 12, if we define* $\Phi, \Phi' : \mathcal{F}(x_1, \ldots, x_n, J) \to \mathcal{U}(\mathcal{H})$ *by*

$$\Phi(x_j) = B_j \text{ for all } 1 \leq j \leq n, \quad \Phi(J) = -\mathbb{1}$$

and

$$\Phi'(x_j) = \begin{cases} A_{ij} & \text{any } i \text{ such that } j \in V_i \\ \mathbb{1} & \text{if no such } i \text{ exists} \end{cases}, \quad \Phi'(J) = -\mathbb{1}$$

then (Φ, Φ') *is a* (τ, κ)*-bipartite representation of* $\Gamma_{Mx=c}$ *with respect to* $|\psi\rangle$*, where*

$$\tau = 2\max(r,4)\sqrt{K(\epsilon + 2^{r-1}\delta)} + \binom{\max(r,4)}{2}2^{r+1}\delta$$

and $\kappa = 2^{r+1}\delta$.

Proof. Follows immediately from Proposition 12, Equation (3.5), and the fact that $B_j^2 = \mathbb{1}$. ◀

4 Embedding finitely-presented groups in solution groups

By Theorem 4, every recursive language can be efficiently encoded as the word problem of a finitely-presented group. By Lemma 6, the word problem for finitely-presented groups reduces to the problem of determining whether $J_G = 1$ in finitely-presented groups G over $\mathbb{Z}_2$. By Theorem 11, if $G = \Gamma_{Mx=c}$ is a solution group, then $J_G = 1$ if and only if $\omega^{co}(\mathcal{G}_{Mx=c}) = 1$.

The main result of [17] is that the problem of determining whether $J_G = 1$ for general finitely-presented groups G over $\mathbb{Z}_2$ reduces to the problem of determining whether $J_\Gamma = 1$ for solution groups $\Gamma = \Gamma_{Mx=c}$. In this paper, we use the following version of this result:

▶ **Theorem 14** ([17]). *Let $G = \langle S : R\rangle$ be a finitely presented group over $\mathbb{Z}_2$, such that $J_G \in S$, and let $N = |S| + \sum_{r\in R} |r|$ be the size of the presentation. Then there is an $m \times n$ linear system $Mx = c$ and a map $\phi : \mathcal{F}(S) \to \mathcal{F}(x_1, \ldots, x_n, J)$ such that*

(a) *$\phi(J_G) = J_\Gamma$, and ϕ descends to an injection $G \to \Gamma_{Mx=c}$ (in other words, for all $w \in \mathcal{F}(S)$, $\phi(w)$ is trivial in $\Gamma_{Mx=c}$ if and only if w is trivial in G);*

(b) *for all $w \in \mathcal{F}(S)$, $|\phi(w)| \leq 4|w|$, and if w is trivial in G, then $\mathrm{Area}_\Gamma(\phi(w)) = O(N \cdot \mathrm{Area}_G(w))$; and*

(c) *M has exactly three non-zero entries in every row, the dimensions m and n of M are $O(N)$, and M and b can be constructed from $\langle S : R\rangle$ in time polynomial in N.*

Note that if $G = \langle S : R\rangle$ and $G' = \langle S' : R'\rangle$ are finitely-presented groups, and $\phi : \mathcal{F}(S) \to \mathcal{F}(S')$ is a homomorphism which descends to a homomorphism $G \to G'$, then $\mathrm{Area}_{G'}(\phi(w)) = O(\mathrm{Area}_G(w))$, with a constant which depends on G, G', and ϕ. The statement in part (b) of Theorem 14 is stronger, in that the constant is independent of G (so the only dependence on G comes from N).

Proof of Theorem 14. Part (a) is Theorem 3.1 of [17]. For the complexity statements in parts (b) and (c), we need to analyze the construction of M and b, which occurs in Proposition 4.3, Corollary 4.8, and Theorem 5.1 of [17]. For this purpose, suppose that $G = \langle S : R\rangle$ is a finitely presented group over $\mathbb{Z}_2$. For simplicity, we assume that $J_G = J \in S$, and that all relations containing J are of the form $J \cdot r = 1$ for some word $r \in \mathcal{F}(S \setminus \{J\})$. This assumption can always be satisfied by adding an extra generator.

For the first step of the construction, we also need some notation. If $x \in \mathcal{F}(S')$ is equal to the reduced word $s_1^{a_1} \cdots s_k^{a_k}$, where $s_i \in S'$ and $a_i \in \{\pm 1\}$ for all $1 \leq i \leq k$, let $x^+ = s_1 \cdots s_k$. Note that this word is still reduced, and that x and x^+ represent the same element in the group

$$\langle S' : s^2 = 1 \text{ for all } s \in S'\rangle.$$

Now, starting from $G = \langle S : R\rangle$, we take a new set of indeterminates $S' = \{u_s, v_s : s \in S \setminus \{J\}\}$, and define $\phi_1 : \mathcal{F}(S) \to \mathcal{F}(S' \cup \{J\})$ by $\phi_1(s) = u_s v_s u_s v_s$ for all $s \in S \setminus \{J\}$ and $\phi_1(J) = J$. We then let

$$G' = \langle S' \cup \{J\} : R' \cup \{u_s^2 = v_s^2 = 1 : s \in S \setminus \{J\}\} \cup \{J^2 = 1\}\rangle,$$

where $R' = \{\phi_1(r)^+ : r \in R\}$. Since $u_s^2 = v_s^2 = J^2 = 1$ in G', we conclude that ϕ_1 descends to a homomorphism $\phi_1 : G \to G'$. It is not hard to see that this morphism is injective (see, for instance, [17, Proposition 4.3]), and clearly $|\phi_1(w)| \leq 4|w|$. If $r \in R$, then $\phi_1(r)$ can be turned into $\phi_1(r)^+$ in at most $4|r|$ applications of the relations $u_s^2 = v_s^2 = 1$, $s \in S \setminus \{J\}$. (In particular, $\mathrm{Area}_{G'}(\phi_1(w)) \leq 4N \,\mathrm{Area}_G(w)$, although we use a more refined calculation for bound on Area_Γ in part (b).) The size of the presentation of G' is

$$N' = |S'| + 1 + \sum_{r\in R'} |r| + 4|S| - 2 \leq 6|S| + 4\sum_{r\in R} |r| \leq 6N,$$

and the presentation can be constructed from $\langle S : R\rangle$ in $O(N)$ time.

To finish the construction of $Mx = c$, we apply the wagon wheel construction from Section 5 of [17] to the group G'. This construction is best understood pictorially. An $m \times n$ matrix M with entries in $\mathbb{Z}_2$ can be represented graphically by drawing a hypergraph with a vertex for each row of M, and an edge for each column, such that the jth hyperedge is incident to the ith vertex if and only if $M_{ij} = 1$. With this representation, a vector $b \in \mathbb{Z}_2^m$ is

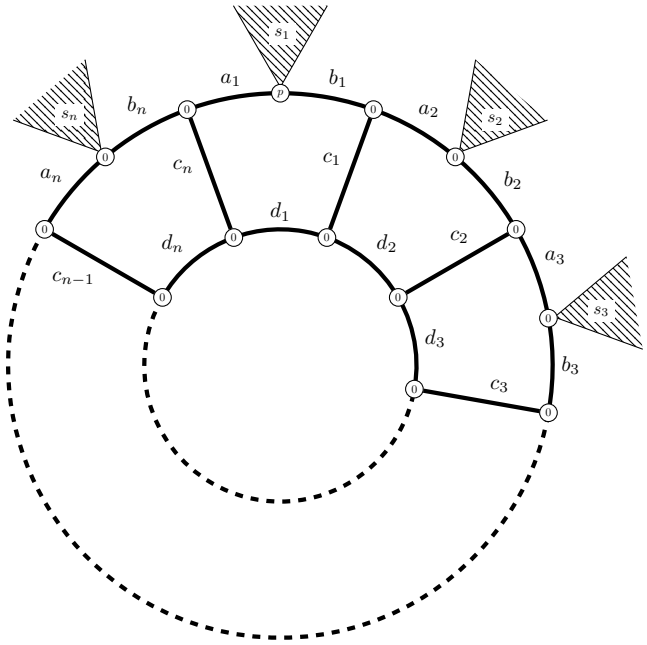


Figure 1 Pictorial depiction of the linear system associated to each relation in the wagon wheel embedding as described in the proof of Theorem 14. Figure reproduced from [17, Figure 2].

the same as function from the vertices of the hypergraph to $\mathbb{Z}_2$. So a linear system $Mx = c$ can thus be represented by a hypergraph with a (not necessarily proper) $\mathbb{Z}_2$-vertex colouring, where the edges correspond to the variables of the system, and the vertices to the equations.

In the wagon wheel construction, $Mx = c$ is defined as a union of subsystems $M^r x^r = c^r$, each corresponding to a relation $r \in R'$. The variables of $Mx = c$ consist of the indeterminates S', as well as an additional set of ancillary variables S''. Each ancillary variable appears in exactly one of the subsystems $M^r x^r = c^r$, while the variables S' are shared. If $r = J^p s_1 \cdots s_n$, where $p \in \mathbb{Z}_2$ and $s_1, \ldots, s_n \in S'$, then the portion of the hypergraph of $Mx = c$ corresponding to $M^r x^r = c^r$ is shown in Figure 1, with the ancillary variables denoted by a_i, b_i, c_i, d_i, $1 \leq i \leq n$. The vertex colouring is also shown in Figure 1: one vertex is given colour p, and the remaining vertices are coloured 0.

As can be seen from Figure 1, the number of ancillary variables added for subsystem $M^r x^r = c^r$ is $4|r|$, and the number of equations added is $3|r|$. Since every vertex in the hypergraph has degree three, every row of M^r has exactly three non-zero entries. Theorem 5.1 of [17] then states that the natural inclusion $\phi_2 : \mathcal{F}(S' \cup \{J\}) \to \mathcal{F}(S' \cup \{J\} \cup S'') : s \mapsto s$ descends to an injection $G' \to \Gamma_{Mx=c}$.

Recall from Definition 10 that every linear equation in $Mx = c$ becomes a defining relation of $\Gamma := \Gamma_{Mx=c}$. The wagon wheel construction is designed so that if $r \in R'$, then $\phi_2(r)$ can be turned into the identity by applying each defining relation from $M_r x = c_r$ exactly once, so $\mathrm{Area}_\Gamma(\phi_2(r)) \leq 3|r|$ for all $r \in R'$. This is easiest to see using pictures of the group, for which we refer to Section 7 of [17]; with this formalism, Figure 1 is itself a proof that $\phi_2(r) = 1$, with each vertex corresponding to a use of the corresponding relation. For relations $r = J^p s_1 \cdots s_n$ with $p \neq 1$, we start with the relation coloured by p, after which J no longer appears in the word. If $r \in R$, then $\phi_2(\phi_1(r))$ can be turned into the identity with at most $7|r|$ applications of the relations of Γ, by first changing $\phi_2(\phi_1(r))$ to $\phi_2(\phi_1(r)^+)$ using the relations $s^2 = 1$, $s \in S'$, and then applying the linear relations of Γ. It follows

that $\mathrm{Area}_\Gamma(\phi_2(w)) = O(N \mathrm{Area}_G(w))$ for all $w \in \mathcal{F}(S)$ which are trivial in G. It should also be clear from Figure 1 that $M^r x^r = c^r$ can be constructed in time polynomial in $|r|$. We conclude that $Mx = c$ is an $m \times n$ linear system with m and n equal to $O(N)$, and that M and b can be constructed in time polynomial in N, so the theorem holds with $\phi = \phi_2 \circ \phi_1$. ◀

Theorem 14 is sufficient to prove Theorem 1. However, to get perfect zero-knowledge protocols for MIP^{co}_δ, we need to prove an additional fact about the embedding in Theorem 14.

▶ **Lemma 15.** *Let $Mx = c$ be an $m \times n$ linear system from the wagon wheel construction in the proof of Theorem 14. In the solution group $\Gamma_{Mx=c}$, the generator x_i is not equal to 1 or J for all $1 \leq i \leq n$, and similarly the product $x_i x_j$ is not equal to 1 or J for all $1 \leq i \neq j \leq n$.*

Proof. We revisit the wagon wheel construction in the proof of Theorem 14. We need to show that $x_i \neq 1$ and $x_i \neq x_j$ in $\Gamma_0 := \Gamma_{Mx=c}/\langle J \rangle$ for all $1 \leq i \neq j \leq n$. This is the same as showing that $x_i \neq 1$ and $x_i \neq x_j$ in $\Gamma_{Mx=0} = \Gamma_0 \times \mathbb{Z}_2$. Recall that the generators of $\Gamma_{Mx=0}$ are split into two sets, the generators S' of G', and the ancillary variables S''. The group $G'_0 := G'/\langle J \rangle$ has a presentation where every generator $s \in S'$ occurs an even number of times in every relation. Thus for any $s \in S'$, we can define a representation $G' \to \mathbb{C}^x$ by sending $s \in S'$ to -1, and $t \in S' \setminus \{s\}$ to 1. It follows that $s \neq 1$ and $s \neq t$ in G'_0 for every $s \neq t \in S'$. Since $G'_0 \to \Gamma_0$ is an injection, we conclude that the same holds in Γ_0.

For the ancillary variables, consider the hypergraph description of the system $Mx = 0$. Given a subset of edges C, let $y \in \mathbb{Z}_2^n$ be the vector with $y_i = 1$ if and only if the ith edge is in C. Then y is a classical solution to $Mx = 0$ if and only if every vertex of the hypergraph is incident with an even number of edges from C. The classical solutions of $Mx = 0$ correspond to 1-dimensional representations of Γ_0; if y is a solution of $Mx = 0$, then the corresponding 1-dimensional representation of Γ_0 sends $x_i \mapsto (-1)^{y_i}$.

Inspecting the wagon wheel hypergraph in Figure 1, we see that every ancilla variable $s \in S''$ belongs to a cycle C which does not contain any edges from S'. Using the corresponding representation of Γ_0, we see that $s \neq 1$ and $s \neq t$ in Γ_0 for all $s \in S''$ and $t \in S'$. Similarly, if $s \neq t \in S''$, and $\{s, t\}$ is not one of the pairs $\{a_i, b_i\}$, then there is a cycle C containing s and not containing t, so $s \neq t$ in Γ_0.

For the pairs $\{a_i, b_i\}$, fix $s \in S''$, and recall that if $r = s_1 \cdots s_n$ is a relation of G', where $s_1, \ldots, s_n \in S'$, then s occurs an even number of times in r. Let $1 \leq i_1 < \cdots < i_{2k} \leq n$ be the indices such that $s_{i_j} = s$, and let

$$\begin{aligned} C_r := \{&s_{i_1}, b_{i_1}, a_{i_1+1}, b_{i_1+1}, \ldots, a_{i_2}, s_{i_2}, \\ &s_{i_3}, b_{i_3}, a_{i_3+1}, b_{i_3+1}, \ldots, a_{i_4}, s_{i_4}, \\ &\ldots, \\ &s_{i_{2k-1}}, b_{i_{2k-1}}, a_{i_{2k-1}+1}, b_{i_{2k-1}+1}, \ldots, a_{i_{2k}}, s_{i_{2k}}\}. \end{aligned}$$

be the collection of paths along the outer cycle of the wagon wheel connection s_{i_1} with s_{i_2}, s_{i_3} with s_{i_4}, and so on. Let $C := \bigcup_{r \in R'} C_r$. Then every vertex of the hypergraph of $Mx = 0$ is incident to an even number of edges in C. If we look at a particular relation r, then for every $1 \leq j \leq 2k$, exactly one of the edges a_{i_j}, b_{i_j} belongs to C_r, so $a_{i_j} \neq b_{i_j}$ in Γ_0. It follows that all of the pairs of ancillary generators a_i, b_i are distinct in Γ_0. ◀

▶ **Proposition 16.** *Let $Mx = c$ be an $m \times n$ linear system from the wagon wheel construction in the proof of Theorem 14, and suppose $J \neq 1$ in $\Gamma_{Mx=c}$. Then $\mathcal{G}_{Mx=c}$ has a commuting-operator strategy $\mathcal{S} = (\mathcal{H}, \{P_a^i\}_{a \in \mathcal{O}_a^i}, \{Q_b^j\}_{b \in \mathbb{Z}_2}, |\psi\rangle)$ such that $\omega(\mathcal{G}_{Mx=c}; \mathcal{S}) = 1$, and*

$$\langle\psi| P_a^i Q_b^j |\psi\rangle = \begin{cases} \frac{1+(-1)^{a_j+b}}{8} & j \in V_i \\ \frac{1}{8} & j \notin V_i \end{cases}$$

for all $1 \leq i \leq m$, $a \in \mathcal{O}_A^i$, $1 \leq j \leq m$, $b \in \mathbb{Z}_2$.

Proof. Suppose $J \neq 1$ in $\Gamma_{Mx=c}$. We recall the construction of a perfect commuting-operator strategy for $\mathcal{G}_{Mx=c}$ from [3]. Let $\mathcal{H} = \ell^2\Gamma_{Mx=c}$ be the regular representation of $\Gamma_{Mx=c}$, and given $g \in \Gamma_{Mx=c}$, let $L(g)$ (resp. $R(g)$) denote left (resp. right) multiplication by g. Then $L(g)$ and $R(g)$ are unitaries for all $g \in \Gamma_{Mx=c}$, and we can get a perfect strategy for $\mathcal{G}_{Mx=c}$ by taking $A_{ij} = L(X_j)$ for all $1 \leq i \leq m$, $j \in V_i$, $B_j = R(X_j)$ for all $1 \leq j \leq n$, and $|\psi\rangle = \frac{1-J}{\sqrt{2}}$ considered as an element of $\mathcal{H}$. Since J is central of order 2, we have that

$$\langle\psi| A_{ik}B_j |\psi\rangle = \langle\psi| L(X_k)R(X_j) |\psi\rangle = \langle\psi| R(X_kX_j) |\psi\rangle = \begin{cases} 1 & X_kX_j = 1 \\ -1 & X_kX_j = J \\ 0 & \text{otherwise.} \end{cases}.$$

Recall from Equation (3.6) that

$$P_a^i = \prod_{k \in V_i} \left(\frac{\mathbb{1} + (-1)^{a_k} A_{ik}}{2} \right)$$

for all $a \in \mathcal{O}_A^i$ and $Q_b^j = \frac{1+(-1)^b B_j}{2}$ for all $b \in \mathcal{O}_B^j$. Using the fact that $\prod_{k \in V_i} A_{ik} = (-\mathbb{1})^{c_i}$ in perfect strategies, and that $|V_i| = 3$ in the linear system constructed in Theorem 14, we get that

$$P_a^i = \prod_{k \in V_i} \left(\frac{\mathbb{1} + (-1)^{a_k} A_{ik}}{2} \right) = \frac{\mathbb{1}}{4} + \frac{1}{4} \sum_{k \in V_i} (-1)^{a_k} A_{ik}.$$

By Lemma 15

$$\langle\psi| P_a^i Q_b^j |\psi\rangle = \frac{1}{8} + \frac{1}{8} \langle\psi| \sum_{k \in V_i} (-1)^{a_k+b} A_{ik}B_j |\psi\rangle = \begin{cases} \frac{1}{8} & j \notin V_i \\ \frac{1+(-1)^{a_j+b}}{8} & j \in V_i. \end{cases}. \quad \blacktriangleleft$$

5 Proof of Theorem 1

In this section we prove Theorem 1, by proving the main technical result of the paper.

▶ **Theorem 17.** *Let $L \subset A^*$ be a language over a finite alphabet A, and contained in* NTIME$(T(n))$*, where $T(n)^4$ is superadditive. Then for any string $w \in A^*$, there is a non-local game $\mathcal{G}_w$ such that*

(a) *the game $\mathcal{G}_w$ has question sets of size $O(|w|)$ and output sets of size at most 8,*
(b) *the function $w \mapsto \mathcal{G}_w$ is computable in $O(|w|^k)$-time, where k is some universal constant,*
(c) *if $w \notin L$ then $\omega^{co}(\mathcal{G}_w) = 1$, and*
(d) *if $w \in L$ then*

$$\omega_\delta^{co}(\mathcal{G}_w) \leq 1 - \frac{1}{T(O(|w|))^{k'}} + O(\delta)$$

for some universal constant k'.

While the constants k, k' in Theorem 17 are independent of L, the other constants appearing in the big-O can depend on L. The game $\mathcal{G}_w$ will be a linear system game $\mathcal{G}_{M(w)x=c(w)}$, where $M(w)x = c(w)$ is an $O(|w|) \times O(|w|)$-linear system. Since the linear system game of an $m \times n$ linear system $Mx = c$ can be constructed in $O(mn)$-time from M and b, the goal in proving Theorem 17 will be to show that the linear system $M(w)x = c(w)$ can be constructed in time polynomial in $|w|$. Theorem 1 is an immediate corollary of Theorem 17.

Proof of Theorem 17. Given the language L, let $G = \langle S : R \rangle$ be the group from Theorem 4, and let κ be the function $A^* \to \mathcal{F}(S)$. Given $w \in A^*$, we let $\widetilde{G}_{\kappa(w)}$ be the group over $\mathbb{Z}_2$ constructed in Definition 5, and $M(w)x = c(w)$ be the linear system constructed from $\widetilde{G}_{\kappa(w)}$ in Theorem 14. Finally, we let $\mathcal{G}_w := \mathcal{G}_{M(w)x=c(w)}$ and $\Gamma_w := \Gamma_{M(w)x=c(w)}$. The only part of the presentation of $\widetilde{G}_{\kappa(w)}$ that changes with w is the relation $[t, [x, \kappa(w)]] = 1$, so the presentation of $\widetilde{G}_{\kappa(w)}$ has size $O(|\kappa(w)|) = O(|w|)$, and $M(w)x = c(w)$ is an $O(|w|) \times O(|w|)$ linear system. Because $M(w)$ has only three non-zero entries per equation, Alice's output sets in $\mathcal{G}_w$ will have size $2^3 = 8$, while Bob's output sets will have size 2. Thus parts (a) and (b) of Theorem 17 follow from part (c) of Theorem 14.

By Theorem 4 and Lemma 6, if $w \notin L$ then $\kappa(w) \neq 1$ in G, and hence $J \neq 1$ in $\widetilde{G}_{\kappa(w)}$. Since the inclusion $\widetilde{G}_{\kappa(w)} \hookrightarrow \Gamma_w$ sends $J_{\widetilde{G}_{\kappa(w)}} \mapsto J_{\Gamma_w}$, we conclude that $J \neq 1$ in Γ_w. By Theorem 11, $\omega^{co}(\mathcal{G}_w) = 1$, proving part (c).

This leaves part (d). Suppose $w \in L$. Then $\kappa(w) = 1$ in G, $J = 1$ in $\widetilde{G}_{\kappa(w)}$, and hence $J = 1$ in Γ_w. Suppose $\mathcal{S}$ is a δ-AC operator strategy for $\mathcal{G}_w$ with $\omega(\mathcal{G}_w; \mathcal{S}) \geq 1 - \epsilon$. Since M has only three non-zero entries per row, the parameters r and K appearing in Corollary 13 are $O(1)$ and $O(|w|)$ respectively. Also, because we are interested in $\delta \leq 2$, we can say that $\delta = O(\sqrt{\delta})$. Thus Corollary 13 states that there is a $(O(\sqrt{|w|(\epsilon + \delta)}), O(\delta))$-bipartite representation (Φ, Φ') of $\mathcal{G}_w$ with respect to the state $|\psi\rangle$ used in $\mathcal{S}$. By construction, this bipartite representation has $\Phi(J) = -\mathbb{1}$. The length of the longest relation in Γ_w is 4, and the length of J in Γ_w is 1, so Lemma 8 implies that

$$-|\psi\rangle = \Phi(J)|\psi\rangle \approx_{O\left(\mathrm{Area}_{\Gamma_w}(J)^2 \sqrt{|w|(\epsilon+\delta)}\right)} |\psi\rangle . \tag{5.1}$$

By Theorem 14, part (b) and Lemma 6, part (c),

$$\mathrm{Area}_{\Gamma_w}(J) = O\left(|w| \cdot \mathrm{Area}_{\widetilde{G}_{\kappa(w)}}(J)\right) = O\left(|w| \cdot \mathrm{Area}_G(\kappa(w))\right).$$

Finally, by Theorem 4, $|\kappa(u)| = O(|u|)$ and Dehn_G is bounded by a function equivalent to $T(n)^4$, so there is a constant C such that $\mathrm{Area}_G(\kappa(w)) = O(T(C|w|)^4 + |w|)$. Since $T(n)^4$ is superadditive by assumption, $|w| = O(T(|w|)^4)$, and we can conclude that $\mathrm{Area}_{\Gamma_w}(J) = O(T(C|w|)^8)$. Returning to Equation (5.1), since $\|-|\psi\rangle - |\psi\rangle\| = 2$, we see that there is a constant $C_0 > 0$ such that

$$C_0 T(C|w|)^{18} \sqrt{\epsilon + \delta} \geq 2.$$

Hence

$$C_0^2 T(C|w|)^{36} (\epsilon + \delta) \geq 4.$$

so,

$$\epsilon \geq \frac{4}{C_0^2 T(C|w|)^{36}} - \delta,$$

So we conclude that

$$\omega(\mathcal{G}_w;\mathcal{S}) \leq 1 - \Omega\left(\frac{1}{T(C|w|)^{36}}\right) + O(\delta).$$

Because $T(n)^4$ is superadditive, $T(C_0 \cdot C|w|)^{36} \geq C_0 T(C|w|)^{36}$ for any integer C_0, so we can move the constant from the big-Ω inside T, proving part (d). ◀

6 Multi-prover interactive proofs

In this section we define the complexity class PZK-MIP$^{co}_{\delta}(2,1,1,1-1/f(n))$, and prove Theorem 2. We first recall the definition of MIP$^{co}_{\delta}$. The definition given here is a simple variant on Definition 8 of [5].

▶ **Definition 18.** *A language L over an alphabet A is in the class* MIP$^{co}_{\delta}(2,1,1,1-1/f(n))$ *of multi-prover interactive proofs with two provers, one round, completness probability* 1*, and soundness probability* $1-1/f(n)$*, if and only if there is family of two-player non-local games* $\mathcal{G}_w = (\mathcal{I}_A^w, \mathcal{I}_B^w, \mathcal{O}_A^{*,w}, \mathcal{O}_B^{*,w}, V_w, \pi_w)$ *indexed by strings* $w \in A^*$*, such that*

- *the input sets* $\mathcal{I}_A^w$*,* $\mathcal{I}_B^w$ *and output sets* $\mathcal{O}_A^{*,w}$*,* $\mathcal{O}_B^{*,w}$ *for* $\mathcal{G}_w$ *are subsets of strings of length* poly $|w|$ *(and hence can have size at most* $2^{\text{poly}\,|w|}$*).*
- *the function* V_w *can be computed in polynomial time in* $|w|$ *and the lengths of its inputs,*
- *the distribution* π_w *can be sampled in polynomial time in* $|w|$ *and the lengths of its inputs,*
- *(completeness) if* $w \in L$ *then* $\omega^{co}(\mathcal{G}_w) = 1$*, and*
- *(soundness) if* $w \notin L$ *then* $\omega^{co}_{\delta}(\mathcal{G}_w) \leq 1 - 1/f(|w|)$*.*

The family $\{\mathcal{G}_w\}$ *is referred to as a* protocol *for* L*.*

Here δ can also be a function of $|w|$. When $\delta = 0$, MIP$^{co}_0$ is the class of commuting-operator multi-prover interactive proofs, which dates back to [9]. Note that, in Definition 18, the protocol must be sound against δ-AC operator strategies, whereas the completeness condition requires a perfect commuting-operator strategy. As a result, MIP$^{co}_{\delta} \subset$ MIP^{co} for all δ. Similarly, MIP$^*_{\delta} \subset$ MIP*.

▶ **Remark 19.** *Our definition is slightly different from [5] in that we use δ-AC strategies with projective measurements, rather than POVMs. It's not clear how this changes the complexity class in general, since we are restricting the class of strategies that a protocol must be sound against (which potentially strengthens the class) and restricting the class of strategies that can be used for completeness (which potentially weakens the class). However, Claim 9 of [5] shows that projective measurements and POVMs are equivalent up to an increase in δ proportional to the size of the output sets. Since our lower bounds use protocols with a constant number of outputs, the lower bounds will also apply if we define* MIP$^{co}_{\delta}$ *using POVMs.*

Next we will define the perfect zero knowledge version of MIP$^{co}_{\delta}$, called PZK-MIP$^{co}_{\delta}$. Informally, a multi-prover interactive proof is perfect zero-knowledge if the verifier gains no new information from interacting with the provers. This is formalized by requiring that, for every yes instance, the provers have a strategy for which the verifier can efficiently simulate the provers' behaviour.

Let $\mathcal{G} = (\mathcal{I}_A, \mathcal{I}_B, \mathcal{O}_A^*, \mathcal{O}_B^*, V, \pi)$ be a non-local game. If the players use a commuting-operator strategy given by measurements $\{P_a^x\}$ and $\{Q_b^y\}$ and a state $|\psi\rangle$ in a Hilbert space $\mathcal{H}$, then to an outside party (such as the verifier), the players actions are described by the probabilities

$$p(a,b|x,y) = \langle\psi| P_a^x Q_b^y |\psi\rangle .$$

When x, y are fixed, $p(a, b|x, y)$ gives a probability distribution over outcomes $(a, b) \in \mathcal{O}_A^x \times \mathcal{O}_B^y$. The family of probability distributions $\underline{p} = \{p(a, b|x, y) : (x, y) \in \mathcal{I}_A \times \mathcal{I}_B, (a, b) \in \mathcal{O}_A^x \times \mathcal{O}_B^y\}$ is called the *correlation matrix* of the strategy.

In a interactive proof system, a record of interactions between verifier and provers is called a transcript. Let $\{\mathcal{G}_w\}$ be a $\mathrm{MIP}_\delta^{co}(2, 1, 1, s)$ protocol for a language L as in Definition 18. During the game $\mathcal{G}_w$, the transcript consists simply of the inputs $(x, y) \in \mathcal{I}_A^w \times \mathcal{I}_B^w$ sent to the provers, and the outputs $(a, b) \in \mathcal{O}_A^x \times \mathcal{O}_B^y$ received back. If the verifier asks questions x, y with probability $\pi(x, y)$, then the distribution over transcripts (x, y, a, b) is given by $\pi(x, y)p(a, b|x, y)$, where $\{p(a, b|x, y)\}$ is the correlation matrix of the provers' strategy. A strategy is said to be *perfect zero-knowledge against an honest verifier* if it is possible to sample from the distribution $\{\pi_w(x, y)p(a, b|x, y)\}_{(x,y,a,b)}$ in polynomial time. However, this assumes that the verifier chooses questions x, y according to the probability distribution π_w given in the protocol, something that the provers cannot validate themselves while the game is in progress. To be perfect zero-knowledge against a possibly dishonest verifier, it is necessary that the verifier be able to simulate $\pi(x, y)p(a, b|x, y)$ for any (simulable) distribution $\pi(x, y)$ on inputs. This is equivalent to being able to simulate the distributions $\{p(a, b|x, y)\}$, so we make the following definition:

▶ **Definition 20.** *Let $\{\mathcal{G}_w\}$ be a $\mathrm{MIP}_\delta^{co}(2, 1, 1, 1 - s)$-protocol for a language L. Then $\{\mathcal{G}_w\}$ is said to be* perfect zero-knowledge *if for each string w and pair $(x, y) \in \mathcal{I}_A \times \mathcal{I}_B$, there is a probability distribution $\{p_w(a, b|x, y) : (a, b) \in \mathcal{O}_A^x \times \mathcal{O}_B^y\}$ over $\mathcal{O}_A^x \times \mathcal{O}_B^y$ such that*

1. *the distribution $\{p_w(a, b|x, y)\}$ can be sampled in polynomial time in $|w|$, $|x|$, and $|y|$, and*
2. *if $w \in L$, then $\{p_w(a, b|x, y) : (x, y) \in \mathcal{I}_A \times \mathcal{I}_B, (a, b) \in \mathcal{O}_A^x \times \mathcal{O}_B^y\}$ is the correlation matrix of a commuting-operator strategy $\mathcal{S}$ with winning probability $\omega(\mathcal{G}_w; \mathcal{S}) = 1$.*

The class PZK-$\mathrm{MIP}_\delta^{co}(2, 1, 1, 1-1/f(n))$ *is the class of languages in* $\mathrm{MIP}_\delta^{co}(2, 1, 1, 1-1/f(n))$ *with a perfect zero-knowledge protocol.*

Proof of Theorem 2. Theorem 17 immediately implies that any language $L \in \mathrm{coNTIME}(f(n))$ has a protocol in $\mathrm{MIP}_\delta^{co}(2, 1, 1, 1 - 1/f(Cn)^k)$ for some constants C and k, where $\delta = o(1/f(Cn)^{2k})$. Since the games constructed in the proof of Theorem 17 come from the wagon wheel construction, Proposition 16 implies that when $w \in L$, the game $\mathcal{G}_w$ has a perfect commuting operator strategy with a correlation that can easily be simulated by the verifier. ◀

6.1 Upper bounds

As mentioned in the introduction, no upper bound on MIP^{co} is known, but an upper bound on MIP_δ^{co} follows from [5] as we will now describe. Consider the following theorem, which is a restatement of Theorem 2 in [5]:

▶ **Theorem** (Theorem 2 [5])**.** *Let G be a 2-prover non-local game with classical messages in which each prover has ℓ possible answers, and $\omega_{QCSDP}^N(G)$ is the optimum of the N-th level of the QC SDP hierarchy for G. Then there exists a $\delta = \Theta(\ell^2/\sqrt{N})$ such that $\omega_\delta^*(G) = \omega_{QCSDP}^N(G)$.*[4]

Here the QC SDP hierarchy for a non-local game G is as in Definition 10 of [5]. For our purposes the only properties of the QC SDP hierarchy that we will require are the following:

[4] The same statement could be made for non-local games with more players by raising the exponent on ℓ.

▶ **Fact 21.** *The QC SDP hierarchy gives an upper bound on the entangled winning probability of a game G at every level. That is, $\omega^N_{QCSDP}(G) \geq \omega^{co}(G)$ for all N. This is an elementary property of this hierarchy and is discussed in [5].*

▶ **Fact 22.** *The quantity $\omega^N_{QCSDP}(G)$ can be computed in time polynomial in $(Q\ell)^N$ where Q is the maximum number of questions to either prover in G, and ℓ is the maximum number of answers. This is because $\omega^N_{QCSDP}(G)$ is defined (in Definition 10 of [5]) to be the optimal value of an semi-definite program on $\mathrm{poly}((Q\ell)^N)$ dimensional space, with $\mathrm{poly}((Q\ell)^N)$ constraints.*

Now, suppose that one wishes to decide whether a non-local game G has $\omega^{co}(G) = 1$, or has $\omega^{co}_\delta(G) \leq 1 - 1/f$ promised that one of the two is the case. By Theorem 2 of [5], there exists $M = O(\ell^4/\delta^2)$ such that $\omega^*_\delta(G) = \omega^M_{QCSDP}(G)$. To resolve the decision problem it then suffices to compute the quantity $\omega^M_{QCSDP}(G)$. In the case that $\omega^{co}(G) = 1$ we know by Fact 21 that we will have $\omega^M_{QCSDP}(G) = 1$. On the other hand, in the case that $\omega^{co}_\delta(G) \leq 1 - 1/f$ we know that $\omega^M_{QCSDP}(G) = \omega^*_\delta(G) \leq \omega^{co}_\delta(G) \leq 1 - 1/f$. It follows by Fact 22 that this decision problem can be solved in time that is polynomial in $(Q\ell)^M = (Q\ell)^{O(\ell^4/\delta^2)}$ where Q and ℓ are the sizes of the question and answer sets in G respectively.

This upper bound uses strategies with POVM measurements, but if we restrict to protocols with constant size output sets, we can state this result for the class defined in Definition 18.

▶ **Theorem 23** ([5])**.** *If $L \in \mathrm{MIP}^{co}_\delta(2, 1, 1, 1 - 1/f(n))$ has a protocol with constant size output sets, and $\delta = o(1/f(n))$, then L is contained in $\mathrm{TIME}(\exp(\mathrm{poly}(n)/\delta^2))$.*

Proof. While the result in [5] is stated for MIP^*_δ which has completeness and soundness conditions stated for finite-dimensional strategies, the proof is still valid for the analogously defined MIP^{co}_δ, which has completeness and soundness conditions stated for commuting-operator strategies. ◀

References

1. Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof Verification and the Hardness of Approximation Problems. *JACM*, 45(3):501–555, 1998.
2. Sanjeev Arora and Shmuel Safra. Probabilistic Checking of Proofs: A New Characterization of NP. *JACM*, 45(1):70–122, 1998.
3. Richard Cleve, Li Liu, and William Slofstra. Perfect commuting-operator strategies for linear system games. *Journal of Mathematical Physics*, 2016. to appear (`arXiv:1606.02278`).
4. Richard Cleve and Rajat Mittal. Characterization of Binary Constraint System Games. In *Automata, Languages, and Programming*, number 8572 in Lecture Notes in Computer Science, pages 320–331. Springer Berlin Heidelberg, 2014. `arXiv:1209.2729`.
5. Matthew Coudron and Thomas Vidick. Interactive proofs with approximately commuting provers. In *International Colloquium on Automata, Languages, and Programming (ICALP 2015)*, pages 355–366, 2015.
6. Andrew C. Doherty, Yeong-Cherng Liang, Benjamin Toner, and Stephanie Wehner. The Quantum Moment Problem and Bounds on Entangled Multi-prover Games. In *Proc. 23rd IEEE Conf. on Computational Complexity (CCC'08)*, pages 199–210. IEEE Computer Society, 2008.
7. J. Fitzsimons, Z. Ji, T. Vidick, and H. Yuen. Quantum proof systems for iterated exponential time, and beyond. *preprint*, 2018. `arXiv:1805.12166`.
8. SM Gersten. Isoperimetric and isodiametric functions of finite presentations. In Graham A. Niblo and Martin A. Roller, editors, *Geometric group theory: Proceedings of the Symposium held in Sussex 1991, Volume 1*, volume 181 of *London Mathematical Society Lecture Note Series 181*, pages 79–97. Cambridge University Press, 1993.

9 T. Ito, H. Kobayashi, D. Preda, X. Sun, and A C.-C. Yao. Generalized Tsirelson Inequalities, Commuting-Operator Provers, and Multi-prover Interactive Proof Systems. In *23rd Annual IEEE Conference on Computational Complexity (CCC 2008)*, 2008.

10 Tsuyoshi Ito and Thomas Vidick. A multi-prover interactive proof for NEXP sound against entangled provers. In *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*, pages 243–252. IEEE, 2012.

11 Zhengfeng Ji. Compression of Quantum Multi-prover Interactive Proofs. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017, pages 289–302, New York, NY, USA, 2017. ACM.

12 R.C. Lyndon and P.E. Schupp. *Combinatorial Group Theory*. Classics in Mathematics. Springer, 1977.

13 Miguel Navascués, Stefano Pironio, and Antonio Acín. Bounding the Set of Quantum Correlations. *Phys. Rev. Lett.*, 98:010401, January 2007. doi:10.1103/PhysRevLett.98.010401.

14 Miguel Navascués, Stefano Pironio, and Antonio Acín. A Convergent Hierarchy of Semidefinite Programs Characterizing the Set of Quantum Correlations. *New Journal of Physics*, 10(073013), 2008. arXiv:0803.4290v1.

15 Narutaka Ozawa. Tsirelson's problem and asymptotically commuting unitary matrices. *Journal of Mathematical Physics*, 54(3):032202, 2013.

16 Mark V. Sapir, Jean-Camille Birget, and Eliyahu Rips. Isoperimetric and Isodiametric Functions of Groups. *Annals of Mathematics*, 156(2):345–466, 2002.

17 William Slofstra. Tsirelson's problem and an embedding theorem for groups arising from non-local games. *CoRR*, 2016. preprint. arXiv:1606.03140.

18 William Slofstra and Thomas Vidick. Entanglement in non-local games and the hyperlinear profile of groups. *Annales Henri Poincare*, 19(10):2979–3005, 2018.

Barriers for Fast Matrix Multiplication from Irreversibility

Matthias Christandl
Department of Mathematical Sciences, University of Copenhagen, Universitetsparken 5, 2100 Copenhagen Ø, Denmark
christandl@math.ku.dk

Péter Vrana
Department of Geometry, Budapest University of Technology and Economics, Egry József u. 1., 1111 Budapest, Hungary
MTA-BME Lendület (Momentum) Research group
vranap@math.bme.hu

Jeroen Zuiddam
Institute for Advanced Study, 1 Einstein Drive, Princeton 08540, NJ, USA
jzuiddam@ias.edu

Abstract

Determining the asymptotic algebraic complexity of matrix multiplication, succinctly represented by the matrix multiplication exponent ω, is a central problem in algebraic complexity theory. The best upper bounds on ω, leading to the state-of-the-art $\omega \leq 2.37..$, have been obtained via the laser method of Strassen and its generalization by Coppersmith and Winograd. Recent barrier results show limitations for these and related approaches to improve the upper bound on ω.

We introduce a new and more general barrier, providing stronger limitations than in previous work. Concretely, we introduce the notion of "irreversibility" of a tensor and we prove (in some precise sense) that any approach that uses an irreversible tensor in an intermediate step (e.g., as a starting tensor in the laser method) cannot give $\omega = 2$. In quantitative terms, we prove that the best upper bound achievable is lower bounded by two times the irreversibility of the intermediate tensor. The quantum functionals and Strassen support functionals give (so far, the best) lower bounds on irreversibility. We provide lower bounds on the irreversibility of key intermediate tensors, including the small and big Coppersmith–Winograd tensors, that improve limitations shown in previous work. Finally, we discuss barriers on the group-theoretic approach in terms of "monomial" irreversibility.

2012 ACM Subject Classification Theory of computation → Algebraic complexity theory

Keywords and phrases Matrix multiplication exponent, barriers, laser method

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.26

Related Version `https://arxiv.org/abs/1812.06952`

Funding *Matthias Christandl*: European Research Council (ERC Grant Agreement No. 337603) and VILLUM FONDEN via the QMATH Centre of Excellence (Grant No. 10059).
Péter Vrana: National Research, Development and Innovation Fund of Hungary within the Quantum Technology National Excellence Program (Project Nr. 2017-1.2.1-NKP-2017-00001) and research grants K124152, KH129601.
Jeroen Zuiddam: National Science Foundation under Grant No. DMS-1638352.

1 Introduction

1.1 Matrix multiplication

Determining the asymptotic algebraic complexity of matrix multiplication is a central open problem in algebraic complexity theory. Several different methods for constructing fast matrix multiplication algorithms have been developed, but on a high level they typically consist of two parts: an efficient reduction of matrix multiplication to an intermediate problem (some bilinear map, i.e. 3-tensor) and an efficient algorithm for the intermediate problem. Recent results have shown "barriers" for such constructions to yield fast matrix multiplication algorithms [4, 7, 8, 2, 3]. We give a barrier, based on a new notion called irreversibility, that is more general and in some cases stronger than the barriers from previous work.

1.2 Matrix multiplication barriers

The matrix multiplication exponent ω is defined as the infimum over all real numbers β such that any two $n \times n$ matrices can be multiplied with $\mathcal{O}(n^\beta)$ algebraic operations, and thus ω represents the asymptotic algebraic complexity of matrix multiplication. Trivially holds $2 \leq \omega \leq 3$. Strassen published the first non-trivial upper bound $\omega \leq \log_2 7$ in 1969 [20]. In the decades that followed, through the development of several ingenious methods by several people, the upper bound was improved to the state-of-the-art bound $\omega \leq 2.37..$, and the pursuit to prove whether $\omega = 2$ or $\omega > 2$ has been ongoing [12, 19, 27, 17, 10, 11]. As mentioned before, these upper bound methods typically consist of a reduction of matrix multiplication to an intermediate problem and an efficient algorithm for the intermediate problem.

Ambainis et al. [4], for the first time, proved a "barrier" result for some collection of such methods. Namely, they showed that a variety of methods that go via the big Coppersmith–Winograd tensor as an intermediate problem cannot give $\omega = 2$, and in fact not even $\omega \leq 2.30...$ We call any lower bound for all upper bounds on ω that can be obtained by some method, a *barrier* for that method. In general, barriers in the sense of limitations to proof methods have a long history in computational complexity theory and recognizing barriers is a natural step towards finding proof methods that do solve the problem at hand.

Next, Alman and Williams [2, 3] extended the realm of barriers beyond the scope of the Ambainis et al. barrier, to a larger collection of methods. Also Blasiak et al. [7, 8] did a study of barriers, namely of barriers for a subset of the group-theoretic method. Both the Blasiak et al. and the Alman and Williams barriers rely on studying versions of "asymptotic subrank" of an intermediate problem.

We give a barrier that applies more generally than all previous barriers and that is in some cases stronger. Our barrier also relies on studying versions of asymptotic subrank, which together with the notion of asymptotic rank we combine into a single parameter called irreversibility. Our barrier simplifies and generalises previous barriers and connects the barrier literature to some central notions from the framework of Strassen [21, 22, 23, 9]. Alman reported similar independent results [1] shortly after our manuscript appeared on the arxiv.

1.3 Our barrier: intuitive explanation

An intuitive explanation of our barrier is as follows. In the language of tensors, the matrix multiplication exponent ω is the optimal "rate of transformation" from the "unit tensor" to

the "matrix multiplication tensor",

$$\text{unit tensor} \xrightarrow{\omega} \text{matrix multiplication tensor.} \tag{1}$$

The rate of transformation naturally satisfies a triangle inequality and thus upper bounds on ω can be obtained by combining the rate of transformation α_1 from the unit tensor to some intermediate tensor and the rate of transformation α_2 from the intermediate tensor to the matrix multiplication tensor; this is the two-component approach alluded to earlier,

$$\text{unit tensor} \xrightarrow{\alpha_1} \text{intermediate tensor} \xrightarrow{\alpha_2} \text{matrix multiplication tensor.} \tag{2}$$

We define the irreversibility of the intermediate tensor as the necessary "loss" that will occur when transforming the unit tensor to the intermediate tensor followed by transforming the intermediate tensor back to the unit tensor. It is well-know that the transformation rate from the matrix multiplication tensor to the unit tensor is $\frac{1}{2}$, so we can extend (2) to

$$\text{unit tensor} \xrightarrow{\alpha_1} \text{intermediate tensor} \xrightarrow{\alpha_2} \text{matrix multiplication tensor} \xrightarrow{1/2} \text{unit tensor.} \tag{3}$$

We thus see that $\alpha_1\alpha_2$ is directly related to the irreversibility of the intermediate tensor, and hence the irreversibility of the intermediate tensor provides limitations on the upper bounds on ω that can be obtained from (2). In particular, any fixed irreversible intermediate tensor cannot show $\omega = 2$ via (2), since the matrix multiplication tensor is reversible when $\omega = 2$.

1.4 Explicit numerical barriers

To exemplify our barrier we show that the support functionals [23] and quantum functionals [9] give (so far, the best) lower bounds on the irreversibility of the following families of tensors:

- the small Coppersmith–Winograd tensors

$$\mathrm{cw}_q = \sum_{i=1}^{q} e_{0,i,i} + e_{i,0,i} + e_{i,i,0}$$

- the big Coppersmith–Winograd tensors

$$\mathrm{CW}_q = e_{0,0,q+1} + e_{0,q+1,0} + e_{q+1,0,0} + \sum_{i=1}^{q} e_{0,i,i} + e_{i,0,i} + e_{i,i,0}$$

- the reduced polynomial multiplication tensors

$$t_n = \sum_{\substack{i,j,k=0:\\ i+j=k}}^{n-1} e_{i,j,k}$$

which for small parameters lead to the following explicit barriers (Theorem 9 and Section 4.2):

q	cw_q-barrier
2	2
3	2.02..
4	2.06..
5	2.09..
6	2.12..
7	2.15..

q	CW_q-barrier
1	2.16..
2	2.17..
3	2.19..
4	2.20..
5	2.21..
6	2.23..

n	t_n-barrier
1	2.17..
2	2.16..
3	2.15..
4	2.15..
5	2.14..
6	2.14..

Indeed, as suggested by the values in the above tables, the cw_q-barrier and CW_q-barrier increase with q (converging to 3), whereas the t_n-barrier decreases with n (converging to 2).

1.5 Comparison and other applications

Compared to Ambainis, Filmus and Le Gall [4] our barriers are valid for a larger class of approaches (and naturally we obtain lower barriers). Compared to Alman and Williams [3] our barriers are valid for a larger class of approaches but our barriers are also higher. As a variation on our barrier we introduce a "monomial" version. Compared to Blasiak, Church, Cohn, Grochow, Naslund, Sawin and Umans [7], and Blasiak, Church, Cohn, Grochow and Umans [8] our monomial barriers are valid for a class of approaches that includes their STPP approach, and thus we provide a uniform view on the barriers that have appeared in the literature. We have not tried to optimise the barriers that we obtain, but focus instead on introducing the barrier itself. The barrier of Alman stated in [1] is very similar to ours, but makes use of asymptotic slice rank instead of asymptotic subrank. Since asymptotic subrank is at most asymptotic slice rank, our barriers are technically stronger. (It is not known whether asymptotic slice rank and asymptotic subrank are equal in general.)

It will become clear to the reader during the development of our ideas that they not only apply to the problem of fast matrix multiplication, but extend to give barriers for the more general problem of constructing fast rectangular matrix multiplication algorithms or even transformations between arbitrary powers of tensors. Such transformations may represent, for example, asymptotic slocc (stochastic local operations and classical communication) reductions among multipartite quantum states [5, 13, 25, 15].

We define irreversibility in Section 2. In Section 3 we introduce the irreversibility barrier. Finally, in Section 4 we present explicit irreversibility barriers.

2 Irreversibility

We begin by introducing some standard notation and terminology. Then we discuss a useful notion called the relative exponent and we define the irreversibility of a tensor. After that we introduce the monomial versions of these ideas and discuss so-called balanced tensors.

2.1 Standard definitions

We assume familiarity with tensors and with the tensor Kronecker product and direct sum. All our tensors will be 3-tensors over some fixed but arbitrary field $\mathbb{F}$. For two tensors $t \in \mathbb{F}^{n_1} \otimes \mathbb{F}^{n_2} \otimes \mathbb{F}^{n_3}$ and $s \in \mathbb{F}^{m_1} \otimes \mathbb{F}^{m_2} \otimes \mathbb{F}^{m_3}$ we write $t \geq s$ and say t restricts to s if there are linear maps $A_i : \mathbb{F}^{n_i} \to \mathbb{F}^{m_i}$ such that $(A_1, A_2, A_3) \cdot t = s$. For $n \in \mathbb{N}$ we define the diagonal tensor (also called the rank-n unit tensor) $\langle n \rangle \coloneqq \sum_{i=1}^{n} e_{i,i,i} \in \mathbb{F}^n \otimes \mathbb{F}^n \otimes \mathbb{F}^n$. The tensor rank of t is defined as $\mathrm{R}(t) \coloneqq \min\{n \in \mathbb{N} : t \leq \langle n \rangle\}$ (this coincides with the definition that $\mathrm{R}(t)$ is the smallest size of any decomposition of t into a sum of simple tensors) and the subrank of t is defined as $\mathrm{Q}(t) \coloneqq \max\{n \in \mathbb{N} : \langle n \rangle \leq t\}$. The asymptotic rank of t is defined as

$$\underset{\sim}{\mathrm{R}}(t) \coloneqq \lim_{n\to\infty} \mathrm{R}(t^{\otimes n})^{1/n} = \inf_n \mathrm{R}(t^{\otimes n})^{1/n} \tag{4}$$

and the asymptotic subrank of t is defined as

$$\underset{\sim}{\mathrm{Q}}(t) \coloneqq \lim_{n\to\infty} \mathrm{Q}(t^{\otimes n})^{1/n} = \sup_n \mathrm{Q}(t^{\otimes n})^{1/n}. \tag{5}$$

The above limits exist and equal the respective infimum and supremum by Fekete's lemma. For $a, b, c \in \mathbb{N}_{\geq 1}$ the matrix multiplication tensor $\langle a, b, c\rangle$ is defined as

$$\langle a, b, c\rangle \coloneqq \sum_{i=1}^{a}\sum_{j=1}^{b}\sum_{k=1}^{c} e_{(i,j),(j,k),(k,i)} \in (\mathbb{F}^a \otimes \mathbb{F}^b) \otimes (\mathbb{F}^b \otimes \mathbb{F}^c) \otimes (\mathbb{F}^c \otimes \mathbb{F}^a). \tag{6}$$

The matrix multiplication exponent is defined as $\omega \coloneqq \log_2 \underset{\sim}{\mathrm{R}}(\langle 2, 2, 2\rangle)$. The meaning of ω in terms of algorithms is: for any $\varepsilon > 0$ there is an algorithm that for any $n \in \mathbb{N}$ multiplies two $n \times n$ matrices using $\mathcal{O}(n^{\omega+\varepsilon})$ scalar additions and multiplications. The difficulty of determining the asymptotic rank of $\langle 2, 2, 2\rangle$ is to be contrasted with the situation for the asymptotic subrank; to put it in Strassen's words: *Unlike the cynic, who according to Oscar Wilde knows the price of everything and the value of nothing, we can determine the asymptotic value of* $\langle h, h, h\rangle$ *precisely* [22],

$$\underset{\sim}{\mathrm{Q}}(\langle h, h, h\rangle) = h^2. \tag{7}$$

2.2 Relative exponent

For a clean exposition of our barrier we will use the notion of relative exponent, which we will define in this section. This notion is inspired by the notion of rate from information theory and alternatively can be seen as a versatile version of the notion of the asymptotic preorder for tensors of Strassen. In the context of tensors, the relative exponent previously appeared in [28] and [26].

▶ **Assumption 1.** *To avoid irrelevant technicalities, we will from now on, without further mentioning, only consider tensors that are not of the form* $u \otimes v \otimes w$.

▶ **Definition 2.** *For two tensors* $t \in \mathbb{F}^{n_1} \otimes \mathbb{F}^{n_2} \otimes \mathbb{F}^{n_3}$ *and* $s \in \mathbb{F}^{m_1} \otimes \mathbb{F}^{m_2} \otimes \mathbb{F}^{m_3}$ *we define the* relative exponent *from* t *to* s *as*

$$\omega(t, s) \coloneqq \lim_{n\to\infty} \tfrac{1}{n} \min\{m \in \mathbb{N} : t^{\otimes m} \geq s^{\otimes n}\} \tag{8}$$

$$= \sup_n \tfrac{1}{n} \min\{m \in \mathbb{N} : t^{\otimes m} \geq s^{\otimes n}\}. \tag{9}$$

The limit is a supremum by Fekete's lemma. Let us briefly relate the relative exponent to the basic notions and results stated earlier. The reader verifies directly that the identities

$$\omega(\langle 2\rangle, t) = \log_2 \underset{\sim}{\mathrm{R}}(t) \tag{10}$$

$$\omega(t, \langle 2\rangle) = 1/(\log_2 \underset{\sim}{\mathrm{Q}}(t)) \tag{11}$$

hold. By definition of the matrix multiplication exponent ω holds

$$\omega(\langle 2\rangle, \langle 2, 2, 2\rangle) = \omega. \tag{12}$$

We know from (7) that

$$\omega(\langle 2, 2, 2\rangle, \langle 2\rangle) = \tfrac{1}{2}. \tag{13}$$

The relative exponent has the following two basic properties that the reader verifies directly.

▶ **Proposition 3.** *Let* s, t *and* u *be tensors.*

(i) $\omega(t, t) = 1$.

(ii) $\omega(s, t)\,\omega(t, u) \geq \omega(s, u)$ (triangle inequality).

2.3 Irreversibility

Our barrier framework relies crucially on the irreversibility of a tensor, a new notion that we define now.

▶ **Definition 4.** *We define the* irreversibility *of a tensor t as the product of the relative exponent from $\langle 2\rangle$ to t and the relative exponent from t to $\langle 2\rangle$, i.e.*

$$\mathbf{i}(t) \coloneqq \omega(\langle 2\rangle, t)\,\omega(t, \langle 2\rangle). \tag{14}$$

Thus $\mathbf{i}(t)$ measures the extent to which the asymptotic conversion from $\langle 2\rangle$ to t is irreversible, explaining the name. Equivalently, the irreversibility is the ratio of the logarithms of the asymptotic rank and the asymptotic subrank, i.e.

$$\mathbf{i}(t) = \frac{\log_2 \underset{\sim}{\mathrm{R}}(t)}{\log_2 \underset{\sim}{\mathrm{Q}}(t)}. \tag{15}$$

From the basic properties of the relative exponent (Proposition 3) follows directly the inequality $\mathbf{i}(t) = \omega(\langle 2\rangle, t)\,\omega(t, \langle 2\rangle) \geq \omega(\langle 2\rangle, \langle 2\rangle) = 1$.

▶ **Proposition 5.** *For any tensor t holds that*

$$\mathbf{i}(t) \geq 1. \tag{16}$$

▶ **Definition 6.** *Let t be a tensor.*

- *If $\mathbf{i}(t) = 1$, then we say that t is* reversible*.*
- *If $\mathbf{i}(t) > 1$, then we say that t is* irreversible*.*

For example, for any $n \in \mathbb{N}$ the diagonal tensor $\langle n\rangle = \sum_{i=1}^{n} e_{i,i,i}$ is reversible. In fact, any reversible tensor t that we know of is equivalent to $\langle n\rangle$ for some n, in the sense that $\langle n\rangle \leq t \leq \langle n\rangle$.

For the matrix multiplication tensor $\langle 2,2,2\rangle$ we have $2\,\mathbf{i}(\langle 2,2,2\rangle) = \omega$ (using (13)). Thus if $\omega = 2$, then $\langle 2,2,2\rangle$ is reversible (and also any other $\langle n,n,n\rangle$). As we will see in Section 3, this is ultimately the source of our barrier.

Irreversible tensors exist. For example, $W = e_{0,0,1} + e_{0,1,0} + e_{1,0,0}$ is irreversible. Namely, it is well-known that $\log_2 \underset{\sim}{\mathrm{R}}(W) = 1$ and that $\log_2 \underset{\sim}{\mathrm{Q}}(W) = h(1/3) = 0.918..$ [23, Theorem 6.7], so $\mathbf{i}(W) = 1.088.. > 1$. In Section 4 we will compute lower bounds on the irreversibility of the small and big Coppersmith–Winograd tensors (that play a crucial role in the best upper bounds on ω).

2.4 Monomial relative exponent and monomial irreversibility

The following restrained version of relative exponent and irreversibility will be relevant. For two tensors $t \in \mathbb{F}^{n_1} \otimes \mathbb{F}^{n_2} \otimes \mathbb{F}^{n_3}$ and $s \in \mathbb{F}^{m_1} \otimes \mathbb{F}^{m_2} \otimes \mathbb{F}^{m_3}$ we write $t \geq_{\mathrm{M}} s$ and say t *monomially restricts* to s if there are linear maps $A_i : \mathbb{F}^{n_i} \to \mathbb{F}^{m_i}$, the corresponding matrices of which are generalised sub-permutation matrices in the standard basis, such that $(A_1, A_2, A_3) \cdot t = s$ [21, Section 6]. Replacing the preorder $\geq$ by $\geq_{\mathrm{M}}$ in Section 2 gives the notions of monomial subrank Q_{M}, monomial asymptotic subrank $\underset{\sim}{\mathrm{Q}}_{\mathrm{M}}$ and monomial relative exponent ω_{M}. (For simplicity we will use monomial restriction here, but our results will also hold with $\geq_{\mathrm{M}}$ replaced by monomial degeneration $\trianglerighteq_{\mathrm{M}}$ defined in [21, Section 6].) Note that the notions Q_{M} and $\underset{\sim}{\mathrm{Q}}_{\mathrm{M}}$ only depend on the support of the tensor, and not on the particular values of the nonzero coefficients. We define the monomial irreversibility $\mathbf{i}_{\mathrm{M}}(t)$ of t as the

product of the (normal) relative exponent from $\langle 2\rangle$ to t and the monomial relative exponent from t to $\langle 2\rangle$,

$$\mathbf{i}_{\mathrm{M}}(t) \coloneqq \omega(\langle 2\rangle, t)\,\omega_{\mathrm{M}}(t, \langle 2\rangle). \tag{17}$$

Equivalently, we have

$$\mathbf{i}_{\mathrm{M}}(t) = \frac{\log_2 \underset{\sim}{\mathrm{R}}(t)}{\log_2 \underset{\sim}{\mathrm{Q}}_{\mathrm{M}}(t)}. \tag{18}$$

(This notion may depend on the tensor and not only on the support.)

▶ **Proposition 7.** *Let s, t and u be tensors.*

(i) $\omega_{\mathrm{M}}(t,t) = 1$.
(ii) $\omega_{\mathrm{M}}(s,t)\,\omega_{\mathrm{M}}(t,u) \geq \omega_{\mathrm{M}}(s,u)$ (triangle inequality).
(iii) $\omega_{\mathrm{M}}(s,t) \geq \omega(s,t)$.
(iv) $\mathbf{i}_{\mathrm{M}}(t) \geq \mathbf{i}(t)$.

▶ **Definition 8.** *Let t be a tensor.*

- *If $\mathbf{i}_{\mathrm{M}}(t) = 1$, then we say that t is* monomially reversible.
- *If $\mathbf{i}_{\mathrm{M}}(t) > 1$, then we say that t is* monomially irreversible.

There exist tensors that are reversible and monomially irreversible. For example, let C be the structure tensor of the algebra $\mathbb{C}[\mathbb{Z}/3\mathbb{Z}]$ in the natural basis,

$$C = e_{0,0,0} + e_{0,1,1} + e_{1,0,1} + e_{2,0,2} + e_{0,2,2} + e_{1,1,2} + e_{1,2,0} + e_{2,1,0} + e_{2,2,1}. \tag{19}$$

Then we have $\underset{\sim}{\mathrm{R}}(C) = 3$, $\underset{\sim}{\mathrm{Q}}(C) = 3$ and $\underset{\sim}{\mathrm{Q}}_{\mathrm{M}}(C) = 2.75..$ (this is proven in [14, 24], see also [9] for the connection to [23]), so that $\mathbf{i}(C) = 1$ and $\mathbf{i}_{\mathrm{M}}(C) = 1.08..$. We note that C is equivalent to the diagonal tensor $\langle 3\rangle$, see e.g. [9] for the basis transformation that shows this.

With regards to matrix multiplication, the standard construction for (13) in fact shows that

$$\omega_{\mathrm{M}}(\langle 2,2,2\rangle, \langle 2\rangle) = \tfrac{1}{2}. \tag{20}$$

2.5 Balanced tensors

We finish this section with a general comment on upper bounds on irreversibility. A tensor $t \in V_1 \otimes V_2 \otimes V_3$ with $\dim(V_1) = \dim(V_2) = \dim(V_3)$ is called *balanced* if the corresponding maps $t_1 : V_1 \to V_2 \otimes V_3$, $t_2 : V_2 \to V_1 \otimes V_3$ and $t_3 : V_3 \to V_1 \otimes V_2$ (called flattenings) are full-rank and for each $i \in [3]$ there is an element $v \in V_i$ such that $t_i(v)$ has full-rank [22, page 121]. For any tensor space with cubic format $\mathbb{F}^n \otimes \mathbb{F}^n \otimes \mathbb{F}^n$ over an algebraically closed field $\mathbb{F}$, being balanced is a generic condition, i.e. almost all elements in such a space are balanced. Balanced tensors are called 1-generic tensors in [16]. Let $t \in \mathbb{F}^n \otimes \mathbb{F}^n \otimes \mathbb{F}^n$ be balanced. Then [22, Proposition 3.6]

$$\underset{\sim}{\mathrm{R}}(t) \leq n^{\frac{2}{3}\omega} \tag{21}$$

$$\underset{\sim}{\mathrm{Q}}(t) \geq n^{\frac{2}{3}} \tag{22}$$

and so

$$\mathbf{i}(t) \leq \omega. \tag{23}$$

If moreover $\underset{\sim}{\mathrm{R}}(t) = n$, then

$$\mathbf{i}(t) \leq 3/2. \tag{24}$$

3 Barriers through irreversibility

With the new notion of irreversibility available, we present a barrier for approaches to upper bound ω via an intermediate tensor t.

3.1 The irreversibility barrier

For any tensor t the inequality

$$\omega(\langle 2\rangle, t)\,\omega(t, \langle 2,2,2\rangle) \geq \omega \tag{25}$$

holds by the triangle inequality. Any such approach to upper bound ω respects the following barrier in terms of the irreversibility $\mathbf{i}(t)$ of t.

▶ **Theorem 9.** *For any tensor t holds*

$$\omega(\langle 2\rangle, t)\,\omega(t, \langle 2,2,2\rangle) \geq 2\,\mathbf{i}(t). \tag{26}$$

Proof. By the triangle inequality (Proposition 3),

$$\omega(\langle 2\rangle, t)\,\omega(t, \langle 2,2,2\rangle)\,\omega(\langle 2,2,2\rangle, \langle 2\rangle) \geq \omega(\langle 2\rangle, t)\,\omega(t, \langle 2\rangle) = \mathbf{i}(t). \tag{27}$$

Therefore, using the fact $\omega(\langle 2,2,2\rangle, \langle 2\rangle) = \frac{1}{2}$ from (13), we have

$$\omega(\langle 2\rangle, t)\,\omega(t, \langle 2,2,2\rangle) \geq \frac{\mathbf{i}(t)}{\omega(\langle 2,2,2\rangle, \langle 2\rangle)} = 2\,\mathbf{i}(t). \tag{28}$$

This proves the claim. ◀

Theorem 9, in particular, implies that if $\mathbf{i}(t) > 1$, then $\omega(\langle 2\rangle, t)\,\omega(t, \langle 2,2,2\rangle) > 2$, i.e. one cannot prove $\omega = 2$ via any fixed irreversible intermediate tensor. (Of course one can consider sequences of intermediate tensors with irreversibility converging to 1.)

3.2 Better barriers through more structure

Naturally, we should expect that imposing more structure on the approach to upper bound ω leads to stronger barriers. In this section we impose that the final step of the approach is an application of the Schönhage τ-theorem. The Schönhage τ-theorem (Strassen's general version [22]) says that

$$\underset{\sim}{\mathrm{R}}\Big(\bigoplus_{i=1}^{q}\langle a_i, b_i, c_i\rangle\Big) \geq \sum_{i=1}^{q}(a_i b_i c_i)^{\omega/3}. \tag{29}$$

In particular holds

$$\underset{\sim}{\mathrm{R}}\big(\langle q\rangle \otimes \langle a,a,a\rangle\big) \geq q a^{\omega}. \tag{30}$$

Therefore, in the language of rates, for any $\alpha, \beta \in \mathbb{N}$ holds that

$$\omega(\langle 2\rangle, \langle 2\rangle^{\alpha}\langle 2,2,2\rangle^{\beta}) \geq \alpha + \beta\omega \tag{31}$$

that is

$$\frac{\omega(\langle 2\rangle, \langle 2\rangle^{\alpha}\langle 2,2,2\rangle^{\beta}) - \alpha}{\beta} \geq \omega. \tag{32}$$

(Here α corresponds to $\log_2 q$ and β corresponds to $\log_2 a$. For simplicity and concreteness we will consider only integer α and β.) Thus for any tensor t and for any $\alpha, \beta \in \mathbb{N}$ holds that

$$\frac{\omega(\langle 2\rangle, t)\,\omega(t, \langle 2\rangle^{\alpha}\langle 2,2,2\rangle^{\beta}) - \alpha}{\beta} \geq \omega. \tag{33}$$

We prove the following barrier in terms of α, β and the irreversibility $\mathbf{i}(t)$ of t.

▶ **Theorem 10.** *For any tensor t and $\alpha, \beta \in \mathbb{N}$ holds*

$$\frac{\omega(\langle 2\rangle, t)\,\omega(t, \langle 2\rangle^{\alpha}\langle 2,2,2\rangle^{\beta}) - \alpha}{\beta} \geq 2\,\mathbf{i}(t) + \frac{\alpha}{\beta}\big(\mathbf{i}(t) - 1\big) \geq 2\,\mathbf{i}(t). \tag{34}$$

Proof. By the triangle inequality,

$$\omega(\langle 2\rangle, t)\,\omega(t, \langle 2\rangle^{\alpha}\langle 2,2,2\rangle^{\beta})\,\omega(\langle 2\rangle^{\alpha}\langle 2,2,2\rangle^{\beta}, \langle 2\rangle) \geq \omega(\langle 2\rangle, t)\,\omega(t, \langle 2\rangle) = \mathbf{i}(t). \tag{35}$$

Therefore,

$$\omega(\langle 2\rangle, t)\,\omega(t, \langle 2\rangle^{\alpha}\langle 2,2,2\rangle^{\beta}) \geq \frac{\mathbf{i}(t)}{\omega(\langle 2\rangle^{\alpha}\langle 2,2,2\rangle^{\beta}, \langle 2\rangle)} = (\alpha + 2\beta)\,\mathbf{i}(t). \tag{36}$$

Subtracting α, dividing by β and using that $\mathbf{i}(t) - 1 \geq 0$ (Proposition 5) gives the barrier

$$\frac{\omega(\langle 2\rangle, t)\,\omega(t, \langle 2\rangle^{\alpha}\langle 2,2,2\rangle^{\beta}) - \alpha}{\beta} \geq \frac{(\alpha + 2\beta)\,\mathbf{i}(t) - \alpha}{\beta} = 2\,\mathbf{i}(t) + \frac{\alpha}{\beta}(\mathbf{i}(t) - 1) \geq 2\,\mathbf{i}(t). \tag{37}$$

This proves the claim. ◀

As a corollary of the above theorem we present a barrier on any approach of the following form. The Schönhage τ-theorem implies that for any $a, b, c \in \mathbb{N}_{\geq 1}$ and any tensor t holds

$$\frac{\omega(\langle 2\rangle, t)\,\omega(t, \langle 2\rangle^{\alpha}\langle a,b,c\rangle) - \alpha}{\frac{1}{3}\log_2(abc)} \geq \omega. \tag{38}$$

We prove the following barrier in terms of a, b, c, α and the irreversibility of the cyclically symmetrized $\mathrm{cyc}(t) \coloneqq t \otimes ((1,2,3)\cdot t) \otimes ((1,2,3)^2 \cdot t)$.

▶ **Corollary 11.** *For any tensor t and $\alpha \in \mathbb{N}$ and $a, b, c \in \mathbb{N}_{\geq 1}$ holds*

$$\frac{\omega(\langle 2\rangle, t)\,\omega(t, \langle 2\rangle^{\alpha}\langle a,b,c\rangle) - \alpha}{\frac{1}{3}\log_2(abc)} \geq 2\,\mathbf{i}(\mathrm{cyc}(t)) + \frac{\alpha}{\frac{1}{3}\log_2(abc)}(\mathbf{i}(\mathrm{cyc}(t)) - 1) \tag{39}$$

$$\geq 2\,\mathbf{i}(\mathrm{cyc}(t)). \tag{40}$$

One verifies that $\mathbf{i}(t) \geq \mathbf{i}(\mathrm{cyc}(t))$. If t is cyclically symmetric, then $\mathrm{cyc}(t) = t^{\otimes 3}$ and we have the equality $\mathbf{i}(t) = \mathbf{i}(\mathrm{cyc}(t))$.

Proof. One verifies directly that $\omega(\langle 2\rangle, t) \geq \omega(\langle 2\rangle, \mathrm{cyc}(t)^{\frac{1}{3}})$ and

$$\omega(t, \langle 2\rangle^{\alpha}\langle a,b,c\rangle) \geq \omega(\mathrm{cyc}(t)^{\frac{1}{3}}, \langle 2\rangle^{\alpha}\langle 2,2,2\rangle^{\frac{1}{3}\log_2(abc)}). \tag{41}$$

Note that we are using real powers here, which is justified by taking powers of the relevant tensors and taking a limit. Using both inequalities and then applying Theorem 10 gives

$$\frac{\omega(\langle 2\rangle, t)\,\omega(t, \langle 2\rangle^{\alpha}\langle a,b,c\rangle) - \alpha}{\frac{1}{3}\log_2(abc)} \geq \frac{\omega(\langle 2\rangle, \mathrm{cyc}(t))\,\omega(\mathrm{cyc}(t), \langle 2\rangle^{\alpha}\langle 2,2,2\rangle^{\frac{1}{3}\log_2(abc)}) - \alpha}{\frac{1}{3}\log_2(abc)} \tag{42}$$

$$\geq 2\,\mathbf{i}(\mathrm{cyc}(t)). \tag{43}$$

This proves the statement of the theorem. ◀

▶ Remark 12. For cyclically symmetric tensors t our Corollary 11 implies the lower bound

$$\omega_g(t) \geq \omega_u(t) \geq 2\,\mathbf{i}(t), \tag{44}$$

on the parameter ω_g (and the "universal" version ω_u) studied in [3], which is a significant improvement over the barrier

$$\omega_g(t) \geq \frac{3}{\frac{1}{2\,\mathbf{i}(t)} + 1} \tag{45}$$

proven in [3, Theorem IV.1].

3.3 Better barriers through monomial irreversibility

Finally, we impose as an extra constraint that the transformation from the intermediate tensor t to the matrix multiplication tensor happens via monomial restriction (Section 2.4), i.e. we consider the approach

$$\omega(\langle 2\rangle, t)\,\omega_{\mathrm{M}}(t, \langle 2,2,2\rangle) \geq \omega \tag{46}$$

and the more structured approaches

$$\frac{\omega(\langle 2\rangle, t)\,\omega_{\mathrm{M}}(t, \langle 2\rangle^{\alpha}\langle 2,2,2\rangle^{\beta}) - \alpha}{\beta} \geq \omega \tag{47}$$

and

$$\frac{\omega(\langle 2\rangle, t)\,\omega_{\mathrm{M}}(t, \langle 2\rangle^{\alpha}\langle a,b,c\rangle) - \alpha}{\frac{1}{3}\log_2(abc)} \geq \omega. \tag{48}$$

The proofs in the previous sections can be directly adapted to prove:

▶ **Theorem 13.** *For any tensor t holds*

$$\omega(\langle 2\rangle, t)\,\omega_{\mathrm{M}}(t, \langle 2,2,2\rangle) \geq 2\,\mathbf{i}_{\mathrm{M}}(t). \tag{49}$$

▶ **Theorem 14.** *For any tensor t and $\alpha, \beta \in \mathbb{N}$ holds*

$$\frac{\omega(\langle 2\rangle, t)\,\omega_{\mathrm{M}}(t, \langle 2\rangle^{\alpha}\langle 2,2,2\rangle^{\beta}) - \alpha}{\beta} \geq 2\,\mathbf{i}_{\mathrm{M}}(t) + \frac{\alpha}{\beta}\big(\mathbf{i}_{\mathrm{M}}(t) - 1\big) \geq 2\,\mathbf{i}_{\mathrm{M}}(t). \tag{50}$$

▶ **Corollary 15.** *For any tensor t and $\alpha \in \mathbb{N}$ and $a, b, c \in \mathbb{N}_{\geq 1}$ holds*

$$\frac{\omega(\langle 2\rangle, t)\,\omega_{\mathrm{M}}(t, \langle 2\rangle^{\alpha}\langle a,b,c\rangle) - \alpha}{\frac{1}{3}\log_2(abc)} \geq 2\,\mathbf{i}_{\mathrm{M}}(\mathrm{cyc}(t)) + \frac{\alpha}{\frac{1}{3}\log_2(abc)}(\mathbf{i}_{\mathrm{M}}(\mathrm{cyc}(t)) - 1) \tag{51}$$

$$\geq 2\,\mathbf{i}_{\mathrm{M}}(\mathrm{cyc}(t)). \tag{52}$$

4 Explicit irreversibility lower bounds

We have seen how barriers arise from lower bounds on irreversibility. In this section we compute lower bounds on the irreversibility of two well-known intermediate tensors that play a crucial role in the best upper bounds on ω: the small and big Coppersmith–Winograd tensors.

4.1 Irreversibility and the asymptotic spectrum of tensors

We begin with a general discussion of how to compute irreversibility. The asymptotic spectrum of tensors is the set of $\leq$-monotone semiring homomorphisms from the semiring of tensors (with tensor product and direct sum as multiplication and addition) to the nonnegative reals,

$$\Delta = \{F \in \mathrm{Hom}(\{\text{tensors}\}, \mathbb{R}_{\geq 0}) : a \leq b \Rightarrow F(a) \leq F(b)\}. \tag{53}$$

Strassen proves in [22] that $\underset{\sim}{\mathrm{Q}}(t) = \min_{F\in\Delta} F(t)$ and $\underset{\sim}{\mathrm{R}}(t) = \max_{F\in\Delta} F(t)$ and he also proves (implicitly) that $\omega(s,t) = \max_{F\in\Delta} \log_2 F(t)/\log_2 F(s)$. From this we directly obtain:

▶ **Proposition 16.** *Let t be a tensor. Then*

$$\mathbf{i}(t) = \frac{\max_{F\in\Delta} \log F(t)}{\min_{F\in\Delta} \log F(t)}. \tag{54}$$

In an ideal world we would know Δ and use it to compute $\mathbf{i}(t)$ (or better, we would use it to compute ω). In practice we currently only have partial knowledge of Δ. This partial knowledge is easiest to describe in terms of the best known lower bounds on $\underset{\sim}{\mathrm{R}}(t)$ and the best known upper bounds on $\underset{\sim}{\mathrm{Q}}(t)$. The best known lower bounds on $\underset{\sim}{\mathrm{R}}(t)$ are simply the matrix ranks of each of the three flattenings t_1, t_2, t_3 of t as described in Section 2.5. For arbitrary fields, the best general upper bounds on $\underset{\sim}{\mathrm{Q}}(t)$ that we are aware of are the Strassen upper support functionals ζ^θ from [23], which we will define and use in the next section. They relate asymptotically to slice rank via [9]

$$\underset{\sim}{\mathrm{Q}}(t) \leq \limsup_n \mathrm{slicerank}(t^{\otimes n})^{1/n} \leq \min_\theta \zeta^\theta(t). \tag{55}$$

We are not aware of any example for which any of the inequalities in (55) is strict. For oblique tensors [1] the right inequality is an equality [9] and for tight tensors [2] both inequalities are equalities [23]. We thus have:

▶ **Proposition 17.** *Let t be a tensor. Then*

$$\mathbf{i}(t) \geq \frac{\max_i \log_2 \mathrm{R}(t_i)}{\min_\theta \log_2 \zeta^\theta(t)}. \tag{56}$$

For tensors over the complex numbers (i.e. $\mathbb{F} = \mathbb{C}$) we have a deeper understanding of the theory of upper bounds on the asymptotic subrank, via the quantum functionals F^θ introduced in [9]. The quantum functionals satisfy $F^\theta \leq \zeta^\theta$ and their minimum equals the asymptotic slice rank [9], i.e.

$$\underset{\sim}{\mathrm{Q}}(t) \leq \limsup_n \mathrm{slicerank}(t^{\otimes n})^{1/n} = \min_\theta F^\theta(t) \leq \min_\theta \zeta^\theta(t). \tag{57}$$

For free tensors[3] the right inequality in (57) is an equality [9]. We thus have:

▶ **Proposition 18.** *Let t be a tensor over the complex numbers. Then*

$$\mathbf{i}(t) \geq \frac{\max_i \log_2 \mathrm{R}(t_i)}{\min_\theta \log_2 F^\theta(t)}. \tag{58}$$

[1] a tensor $t \in \mathbb{F}^{n_1} \otimes \mathbb{F}^{n_2} \otimes \mathbb{F}^{n_3}$ is called oblique if the support $\mathrm{supp}(t) \in [n_1] \times [n_2] \times [n_3]$ in some basis is an antichain in the product of the natural orders on the $[n_i]$

[2] a tensor t is called tight if for some choice of basis there are injective maps $\alpha_1, \alpha_2, \alpha_3$ such that for every $a \in \mathrm{supp}(t)$ holds $\alpha_1(a_1) + \alpha_2(a_2) + \alpha_3(a_3) = 0$

[3] a tensor t is called free if in some basis any two different $a, b \in \mathrm{supp}(t)$ differ in at least two entries

4.2 Irreversibility of Coppersmith–Winograd tensors

We now compute lower bounds for the irreversibility of the Coppersmith–Winograd tensors. As mentioned, we will use the support functionals of Strassen [23] in our computation to upper bound the asymptotic subrank. For any $\theta \in \mathbb{R}^3_{\geq 0}$ with $\theta_1 + \theta_2 + \theta_3 = 1$ the upper support functional ζ^θ is defined as

$$\zeta^\theta(t) \coloneqq 2^{\rho^\theta(t)} \tag{59}$$

$$\rho^\theta(t) \coloneqq \min_{s \cong t} \max_{P \in \mathcal{P}(\mathrm{supp}(s))} \sum_{i=1}^{3} \theta_i H(P_i), \tag{60}$$

where the minimum is over all tensors s isomorphic to t, the maximum is over all probability distributions on the support of s in the standard basis, and $H(P_i)$ denotes the Shannon entropy of the ith marginal of P. Strassen proves in [23] that $1/\omega(t, \langle 2 \rangle) = \log_2 \underset{\sim}{\mathrm{Q}}(t) \leq \rho^\theta(t)$.

(Besides from the Strassen support functionals, upper bound on the asymptotic subrank of complex tensors may be obtained from the quantum functionals. For the tensors in Theorem 19 and Theorem 22, however, this will give the same bound, since these tensors are free [9, Section 4.3].)

▶ **Theorem 19** (Small Coppersmith–Winograd tensors [12, Section 6])**.** *For the small Coppersmith–Winograd tensor*

$$\mathrm{cw}_q \coloneqq \sum_{i=1}^{q} e_{0,i,i} + e_{i,0,i} + e_{i,i,0} \tag{61}$$

the lower bound

$$2\,\mathbf{i}(\mathrm{cw}_q) \geq \frac{2\log_2(q+1)}{\log_2 3 - \frac{2}{3} + \frac{2}{3}\log_2 q} \tag{62}$$

holds.

Proof. The rank of each flattening of cw_q equals $q+1$. Therefore, $\underset{\sim}{\mathrm{R}}(\mathrm{cw}_q) \geq q+1$. To upper bound the asymptotic subrank $\underset{\sim}{\mathrm{Q}}(\mathrm{cw}_q)$ one can upper bound the Strassen upper support functional with $\theta = (1/3, 1/3, 1/3)$ as in [9, Example 4.22] by

$$\rho^\theta(\mathrm{cw}_q) \leq \log_2 3 - \frac{2}{3} + \frac{2}{3}\log_2 q. \tag{63}$$

We find that

$$\mathbf{i}(\mathrm{cw}_q) \geq \frac{\log_2(q+1)}{\log_2 3 - \frac{2}{3} + \frac{2}{3}\log_2 q}. \tag{64}$$

This proves the theorem. ◀

▶ Remark 20. If $q > 2$, then the right-hand side of (62) is at least 2.02.. See the table in Section 1 for more values. If $q = 2$, however, then the right-hand side of (62) equals 2. Theorem 19 thus does not rule out using cw_2 to prove that $\omega = 2$. Indeed, as observed in [12, Section 11]), if $\omega(\langle 2 \rangle, \mathrm{cw}_2) = \log_2 3$, then $\omega = 2$.

Currently, the best upper bound we have on $\omega(\langle 2 \rangle, \mathrm{cw}_q)$ is $\log_2(q+2)$. If $\omega(\langle 2 \rangle, \mathrm{cw}_q) = \log_2(q+2)$, then instead of (62) we get the better barrier

$$2\,\mathbf{i}(\mathrm{cw}_q) \geq \frac{2\log_2(q+2)}{\log_2 3 - \frac{2}{3} + \frac{2}{3}\log_2 q}. \tag{65}$$

The right-hand side of (65) has a minimum value of

$$\frac{18}{5\log_2 3} = 2.27.. \tag{66}$$

attained at $q = 6$.

▶ Remark 21. The following computation serves as a sanity check for our barrier. Namely we see in an example how by putting some extra assumption the barrier becomes tight. Coppersmith and Winograd in [12] used cw_q as an intermediate tensor in combination with the laser method and a certain "outer structure", see also [6, Section 9]. When we impose that we apply the laser method on cw_q with this outer structure to upper bound ω we get the following better barrier via Theorem 10/Corollary 11:

$$2\,\mathbf{i}(\mathrm{cw}_q) + \frac{h(1/3)}{\frac{1}{3}\log_2(q)}(\mathbf{i}(\mathrm{cw}_q) - 1) \tag{67}$$

where

$$\mathbf{i}(\mathrm{cw}_q) \geq \frac{\log_2(q+1)}{\log_2(3) - \frac{2}{3} + \frac{2}{3}\log_2(q)}. \tag{68}$$

Some values of (67) are:

q	
2	2
3	2.04..
4	2.10..
5	2.15..
6	2.19..
7	2.22..

If in addition we assume that $\omega(\langle 2\rangle, \mathrm{cw}_q) = \log_2(q+2)$, then we obtain the barrier

$$2\,\mathbf{i}(\mathrm{cw}_q) + \frac{h(1/3)}{\frac{1}{3}\log_2(q)}(\mathbf{i}(\mathrm{cw}_q) - 1) \tag{69}$$

where

$$\mathbf{i}(\mathrm{cw}_q) \geq \frac{\log_2(q+2)}{\log_2(3) - \frac{2}{3} + \frac{2}{3}\log_2(q)}. \tag{70}$$

Some values of (69) are:

q	
2	3.24..
3	2.65..
4	2.50..
5	2.44..
6	2.41..
7	2.40..
8	2.40..
9	2.40..
10	2.40..
11	2.41..

with minimum value of 2.40... These barriers in fact match the upper bound

$$\omega \leq \log_q \frac{4(q+2)^3}{27} \tag{71}$$

that was obtained by Coppersmith and Winograd by applying the laser method in the way described above. Other intermediate tensors with a given outer structure may be analyzed similarly.

▶ **Theorem 22** (Big Coppersmith–Winograd tensors [12, Section 7]). *For the big Coppersmith–Winograd tensor*

$$\mathrm{CW}_q \coloneqq e_{0,0,q+1} + e_{0,q+1,0} + e_{q+1,0,0} + \sum_{i=1}^{q} e_{0,i,i} + e_{i,0,i} + e_{i,i,0} \tag{72}$$

the lower bound

$$2\,\mathbf{i}(\mathrm{CW}_q) \geq \begin{cases} \dfrac{2\log_2(3)}{f(\frac{1}{18}(\sqrt{33}-3))} = 2.16.. & q=1 \\ \dfrac{2\log_2(4)}{f(\frac{1}{9})} = 2.17.. & q=2 \\ \dfrac{2\log_2(q+2)}{f\left(\frac{3q-\sqrt{32+q^2}}{6(q^2-4)}\right)} & q \geq 3 \end{cases} \tag{73}$$

holds, where

$$f(x) \coloneqq -\Big(\frac{2}{3}-qx\Big)\log_2\Big(\frac{2}{3}-qx\Big) - q2x\log_2(2x) - \Big(\frac{1}{3}-qx\Big)\log_2\Big(\frac{1}{3}-qx\Big). \tag{74}$$

Proof. The rank of each flattening of CW_q equals $q+2$, which coincides with the well-known border rank upper bound $\underline{\mathrm{R}}(\mathrm{CW}_q) \leq q+2$. Therefore, $\underset{\sim}{\mathrm{R}}(\mathrm{CW}_q) = q+2$.

To upper bound the asymptotic subrank $\underset{\sim}{\mathrm{Q}}(\mathrm{CW}_q)$ we use the Strassen upper support functional with $\theta = (1/3, 1/3, 1/3)$. In the standard basis, the support of CW_q is the set

$$\{(0,i,i),(i,0,i),(i,i,0) : i \in [q]\} \cup \{(0,0,q+1),(0,q+1,0),(q+1,0,0)\}. \tag{75}$$

The symmetry implies that we can assign probability x to each of $(0,i,i)$, $(i,0,i)$ and $(0,i,i)$, and $\frac{1}{3}-qx$ to $(0,0,q+1)$, $(0,q+1,0)$ and $(q+1,0,0)$. This leads to an average marginal entropy of $f(x)$ as defined in the theorem statement. The maximum of $f(x)$ is attained at

$$x = \begin{cases} \frac{1}{18}(\sqrt{33}-3) & q=1 \\ \frac{1}{9} & q=2 \\ \dfrac{3q-\sqrt{32+q^2}}{6(q^2-4)} & q \geq 3. \end{cases} \tag{76}$$

This proves the theorem. ◀

▶ Remark 23. The lowest value of the right-hand side of (73) is 2.16.. attained at $q=1$. See the table in Section 1 for more values.

▶ Remark 24. A lower bound on the irreversibility of the tensors t_n mentioned in the introduction follows directly from the results in [23, Theorem 6.7].

4.3 Monomial irreversibility of structure tensors of finite group algebras

We now discuss irreversibility and monomial irreversibility in the context of the group-theoretic approach developed in [10]. This approach produces upper bounds on ω via intermediate tensors that are structure tensors of complex group algebras of finite groups. Let $\langle G\rangle$ denote the structure tensor of the complex group algebra $\mathbb{C}[G]$ of the finite group G, in the standard basis. The group-theoretic approach (in particular [10, Theorem 4.1]) produces an inequality of the form

$$\langle G\rangle \geq_{\mathrm{M}} \langle a,b,c\rangle \tag{77}$$

which ultimately (see [10, Eq. (1)]) leads to the bound

$$\frac{\omega(\langle 2\rangle,\langle G\rangle)\,\omega_{\mathrm{M}}(\langle G\rangle,\langle a,b,c\rangle)}{\frac{1}{3}\log_2(abc)} \geq \omega \tag{78}$$

where $\geq_{\mathrm{M}}$ and ω_{M} are the monomial restriction and monomial relative exponent defined in Section 2.4.

Now the monomial irreversibility barrier from Section 3.3 comes into play. Upper bounds on the monomial asymptotic subrank of $\langle G\rangle$ have (using different terminology) been obtained in [7, 8, 18]. Those upper bounds imply that $\langle G\rangle$ is monomially irreversible for every nontrivial finite group G. Together with our results in Section 3.3 and the fact that the tensor $\langle G\rangle$ is symmetric up to a permutation of the basis of one of the tensor legs, this directly leads to nontrivial barriers for the left-hand side of (78) for any fixed nontrivial group G, thus putting the work of [7, 8, 18] in a broader context. We have not tried to numerically optimise the monomial irreversibility barriers for group algebras.

Finally we mention that the irreversibility barrier (rather than the monomial irreversibility barrier) does not rule out obtaining $\omega = 2$ via $\langle G\rangle$. Namely, $\langle G\rangle$ is isomorphic to a direct sum of matrix multiplication tensors, $\langle G\rangle \cong \bigoplus_i \langle d_i, d_i, d_i\rangle$ and, therefore, we have $\mathbf{i}(\langle G\rangle) = (\log_2 \sum_i d_i^\omega)/(\log_2 \sum_i d_i^2)$. Thus, if $\omega = 2$, then $\langle G\rangle$ is reversible.

References

1 Josh Alman. Limits on the Universal Method for Matrix Multiplication. *arXiv*, 2018. arXiv:1812.08731.

2 Josh Alman and Virginia Vassilevska Williams. Further Limitations of the Known Approaches for Matrix Multiplication. In Anna R. Karlin, editor, *9th Innovations in Theoretical Computer Science Conference (ITCS 2018)*, volume 94 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 25:1–25:15, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.ITCS.2018.25.

3 Josh Alman and Virginia Vassilevska Williams. Limits on All Known (and Some Unknown) Approaches to Matrix Multiplication. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 580–591, October 2018. doi:10.1109/FOCS.2018.00061.

4 Andris Ambainis, Yuval Filmus, and François Le Gall. Fast matrix multiplication: limitations of the Coppersmith-Winograd method (extended abstract). In *STOC'15—Proceedings of the 2015 ACM Symposium on Theory of Computing*, pages 585–593. ACM, New York, 2015. doi:10.1145/2746539.2746554.

5 Charles H. Bennett, Sandu Popescu, Daniel Rohrlich, John A. Smolin, and Ashish V. Thapliyal. Exact and asymptotic measures of multipartite pure-state entanglement. *Phys. Rev. A*, 63(1):012307, 2000. doi:10.1103/PhysRevA.63.012307.

6 Markus Bläser. *Fast Matrix Multiplication*. Number 5 in Graduate Surveys. Theory of Computing Library, 2013. doi:10.4086/toc.gs.2013.005.

7 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 Anal.*, 2017. doi:10.19086/da.1245.

8 Jonah Blasiak, Thomas Church, Henry Cohn, Joshua A Grochow, and Chris Umans. Which groups are amenable to proving exponent two for matrix multiplication? *arXiv*, 2017. arXiv:1712.02302.

9 Matthias Christandl, Péter Vrana, and Jeroen Zuiddam. Universal points in the asymptotic spectrum of tensors. In *Proceedings of 50th Annual ACM SIGACT Symposium on the Theory of Computing (STOC'18)*. ACM, New York, 2018. doi:10.1145/3188745.3188766.

10 Henry Cohn and Christopher Umans. A group-theoretic approach to fast matrix multiplication. In *Foundations of Computer Science, 2003. Proceedings. 44th Annual IEEE Symposium on*, pages 438–449. IEEE, 2003. doi:10.1109/SFCS.2003.1238217.

11 Henry Cohn and Christopher Umans. Fast matrix multiplication using coherent configurations. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1074–1086. SIAM, 2013. URL: http://dl.acm.org/citation.cfm?id=2627817.2627894.

12 Don Coppersmith and Shmuel Winograd. Matrix multiplication via arithmetic progressions. *J. Symbolic Comput.*, 9(3):251–280, 1990. doi:10.1016/S0747-7171(08)80013-2.

13 Wolfgang Dür, Guivre Vidal, and Juan Ignacio Cirac. Three qubits can be entangled in two inequivalent ways. *Phys. Rev. A (3)*, 62(6):062314, 12, 2000. doi:10.1103/PhysRevA.62.062314.

14 Jordan S. Ellenberg and Dion Gijswijt. On large subsets of $\mathbf{F}_q^n$ with no three-term arithmetic progression. *Ann. of Math. (2)*, 185(1):339–343, 2017. doi:10.4007/annals.2017.185.1.8.

15 Ryszard Horodecki, Paweł Horodecki, Michał Horodecki, and Karol Horodecki. Quantum entanglement. *Rev. Modern Phys.*, 81(2):865–942, 2009. doi:10.1103/RevModPhys.81.865.

16 J.M. Landsberg and Mateusz Michałek. Abelian tensors. *Journal de Mathématiques Pures et Appliquées*, 108(3):333–371, 2017. doi:10.1016/j.matpur.2016.11.004.

17 François Le Gall. Powers of tensors and fast matrix multiplication. In *ISSAC 2014—Proceedings of the 39th International Symposium on Symbolic and Algebraic Computation*, pages 296–303. ACM, New York, 2014. doi:10.1145/2608628.2608664.

18 Will Sawin. Bounds for matchings in nonabelian groups. *arXiv*, 2017. arXiv:1702.00905.

19 Andrew James Stothers. *On the complexity of matrix multiplication*. PhD thesis, University of Edinburgh, 2010. URL: http://hdl.handle.net/1842/4734.

20 Volker Strassen. Gaussian elimination is not optimal. *Numer. Math.*, 13(4):354–356, 1969. doi:10.1007/BF02165411.

21 Volker Strassen. Relative bilinear complexity and matrix multiplication. *J. Reine Angew. Math.*, 375/376:406–443, 1987. doi:10.1515/crll.1987.375-376.406.

22 Volker Strassen. The asymptotic spectrum of tensors. *J. Reine Angew. Math.*, 384:102–152, 1988. doi:10.1515/crll.1988.384.102.

23 Volker Strassen. Degeneration and complexity of bilinear maps: some asymptotic spectra. *J. Reine Angew. Math.*, 413:127–180, 1991. doi:10.1515/crll.1991.413.127.

24 Terence Tao. A symmetric formulation of the Croot–Lev–Pach–Ellenberg–Gijswijt capset bound, 2016. URL: https://terrytao.wordpress.com.

25 F. Verstraete, J. Dehaene, B. De Moor, and H. Verschelde. Four qubits can be entangled in nine different ways. *Phys. Rev. A (3)*, 65(5, part A):052112, 5, 2002. doi:10.1103/PhysRevA.65.052112.

26 Péter Vrana and Matthias Christandl. Asymptotic entanglement transformation between W and GHZ states. *J. Math. Phys.*, 56(2):022204, 12, 2015. doi:10.1063/1.4908106.

27 Virginia Vassilevska Williams. Multiplying matrices faster than Coppersmith-Winograd. Extended abstract. In *STOC'12—Proceedings of the 2012 ACM Symposium on Theory of Computing*, pages 887–898. ACM, New York, 2012. doi:10.1145/2213977.2214056.

28 Nengkun Yu, Cheng Guo, and Runyao Duan. Obtaining a W state from a Greenberger-Horne-Zeilinger state via stochastic local operations and classical communication with a rate approaching unity. *Physical review letters*, 112(16):160401, 2014. doi:10.1103/PhysRevLett.112.160401.

Hardness Magnification near State-Of-The-Art Lower Bounds

Igor Carboni Oliveira
Department of Computer Science, University of Oxford, UK
igor.carboni.oliveira@cs.ox.ac.uk

Ján Pich
Department of Computer Science, University of Oxford, UK
jan.pich@cs.ox.ac.uk

Rahul Santhanam
Department of Computer Science, University of Oxford, UK
rahul.santhanam@cs.ox.ac.uk

Abstract

This work continues the development of hardness magnification. The latter proposes a new strategy for showing strong complexity lower bounds by reducing them to a refined analysis of weaker models, where combinatorial techniques might be successful.

We consider gap versions of the meta-computational problems MKtP and MCSP, where one needs to distinguish instances (strings or truth-tables) of complexity $\leq s_1(N)$ from instances of complexity $\geq s_2(N)$, and $N = 2^n$ denotes the input length. In MCSP, complexity is measured by circuit size, while in MKtP one considers Levin's notion of time-bounded Kolmogorov complexity. (In our results, the parameters $s_1(N)$ and $s_2(N)$ are asymptotically quite close, and the problems almost coincide with their standard formulations without a gap.) We establish that for Gap-MKtP$[s_1, s_2]$ and Gap-MCSP$[s_1, s_2]$, a *marginal improvement* over the state-of-the-art in unconditional lower bounds in a variety of computational models would imply explicit *super-polynomial* lower bounds.

Theorem. There exists a universal constant $c \geq 1$ for which the following hold. If there exists $\varepsilon > 0$ such that for every small enough $\beta > 0$

(1) Gap-MCSP$[2^{\beta n}/cn, 2^{\beta n}] \notin$ Circuit$[N^{1+\varepsilon}]$, then NP $\not\subseteq$ Circuit[poly].

(2) Gap-MKtP$[2^{\beta n}, 2^{\beta n} + cn] \notin \mathsf{TC}^0[N^{1+\varepsilon}]$, then EXP $\not\subseteq \mathsf{TC}^0$[poly].

(3) Gap-MKtP$[2^{\beta n}, 2^{\beta n} + cn] \notin B_2$-Formula$[N^{2+\varepsilon}]$, then EXP $\not\subseteq$ Formula[poly].

(4) Gap-MKtP$[2^{\beta n}, 2^{\beta n} + cn] \notin U_2$-Formula$[N^{3+\varepsilon}]$, then EXP $\not\subseteq$ Formula[poly].

(5) Gap-MKtP$[2^{\beta n}, 2^{\beta n} + cn] \notin$ BP$[N^{2+\varepsilon}]$, then EXP $\not\subseteq$ BP[poly].

(6) Gap-MKtP$[2^{\beta n}, 2^{\beta n} + cn] \notin (\mathsf{AC}^0[6])[N^{1+\varepsilon}]$, then EXP $\not\subseteq \mathsf{AC}^0[6]$.

These results are complemented by lower bounds for Gap-MCSP and Gap-MKtP against different models. For instance, the lower bound assumed in (1) holds for U_2-formulas of near-quadratic size, and lower bounds similar to (3)-(5) hold for various regimes of parameters.

We also identify a natural computational model under which the hardness magnification threshold for Gap-MKtP lies *below* existing lower bounds: U_2-formulas that can compute parity functions at the leaves (instead of just literals). As a consequence, if one managed to adapt the existing lower bound techniques against such formulas to work with Gap-MKtP, then EXP $\not\subseteq \mathsf{NC}^1$ would follow via hardness magnification.

2012 ACM Subject Classification Theory of computation

Keywords and phrases Circuit Complexity, Minimum Circuit Size Problem, Kolmogorov Complexity

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.27

Related Version A preprint of this work appeared at the Electronic Colloquium on Computational Complexity (ECCC) as the report TR 18-158 and is available at `https://eccc.weizmann.ac.il/report/2018/158/`.

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka; Article No. 27; pp. 27:1–27:29

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

Funding This work was supported in part by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2014)/ERC Grant Agreement no. 615075 and by the Austrian Science Fund (FWF) under project number P 28699.

Acknowledgements We thank Avishay Tal for bringing [55] to our attention in connection to the problem of proving lower bounds against U_2-Formula-$\oplus$. We are also grateful to Jan Krajíček for discussions related to hardness magnification.

1 Introduction

1.1 Context

Establishing limits on the efficiency of computations is widely considered to be one of the most important open problems in computer science and mathematics. Unconditional lower bounds are known in many restricted computational settings (see e.g. [8, 24]), but progress in understanding the limitations of more expressive devices has been slow and incremental (cf. [1] for a recent survey and references). Table 1 summarizes the current landscape of unconditional lower bounds with respect to general circuits, formulas, branching programs, bounded-depth threshold circuits, and bounded-depth circuits with modular gates. These constitute some of the most widely investigated models extending the weak computational settings for which we already have explicit super-polynomial lower bounds.

Table 1 A summary of several state-of-the-art lower bounds in circuit complexity theory. In our notation, N denotes input length, and $\mathfrak{C}[s]$ refers to $\mathfrak{C}$-circuits of size $\leq s$. Establishing stronger lower bounds in these different models is open (or non-trivial lower bounds for a function in $\mathsf{E} = \mathsf{DTIME}[2^{O(N)}]$ in the case of ACC^0_d).

Computational Model	Unconditional Lower Bounds	Reference(s)
Boolean Circuits; w.r.t. different forms of explicitness	$\mathsf{P} \not\subseteq \mathsf{Circuit}[cN]$, $\mathsf{MA}/1 \not\subseteq \mathsf{Circuit}[N^k]$ $\mathsf{MA}_{\mathsf{EXP}} \not\subseteq \mathsf{Circuit}[\mathsf{poly}]$	[23, 14] [9, 49]
Formulas over B_2	$\mathsf{P} \not\subseteq B_2\text{-}\mathsf{Formula}[N^{2-o(1)}]$	[39]
Formulas over U_2	$\mathsf{P} \not\subseteq U_2\text{-}\mathsf{Formula}[N^{3-o(1)}]$	[16, 54, 13]
Branching programs	$\mathsf{P} \not\subseteq \mathsf{BP}[N^{2-o(1)}]$	[39]
Low-depth threshold circuits	$\mathsf{P} \not\subseteq \mathsf{MAJ} \circ \mathsf{THR} \circ \mathsf{THR}[N^{3/2-o(1)}]$	[27]
Depth-d threshold circuits	$\mathsf{P} \not\subseteq \mathsf{TC}^0_d[N^{1+\exp(-d)}]$ (wires)	[22]
Depth-d circuits with mod gates	$\mathsf{quasi\text{-}NP} \not\subseteq \mathsf{ACC}^0_d[\mathsf{poly}]$	[38]

A conditional explanation has been proposed to address the difficulty of establishing strong lower bounds in most of these computational settings. The theory of natural proofs [48] shows that if a computational device can compute pseudorandom functions, then sufficiently constructive techniques (such as those that have been successful against weaker models) cannot show lower bounds of the form N^k if k is sufficiently large. This connection has been quite influential, and subsequent works (see e.g. [36, 7]) have further investigated the limitations of lower bound techniques from this perspective.

The Razborov-Rudich framework suggests that proving unconditional lower bounds in stronger computational models might be tightly related to the investigation of meta-computational problems of a particular form: *those referring to the computational complexity*

of strings or truth-tables. Indeed, it has been subsequently proved that the existence of a natural property for a class of circuits yields explicit lower bounds against the same class [20, 60, 40, 19].

Our results describe a striking phenomenon associated to such problems. They show that in several scenarios, if we could establish slightly stronger lower bounds for them, i.e., lower bounds that marginally improve the size bounds described in Table 1, then *super-polynomial* lower bounds for explicit problems would follow. More specifically, this phenomenon concerns computational problems where the complexity of strings are measured according to circuit complexity (often referred to as MCSP; see [26]) or Levin's time-bounded Kolmogorov complexity [32] (a problem known as MKtP; see [5]). MCSP and MKtP are important meta-computational problems with connections to areas such as learning theory, cryptography, proof complexity, pseudorandomness and circuit complexity (see e.g. [4] and references therein). We refer to [3] for more discussion about the importance of these and related complexity measures.

The new results are part of an emerging theory of *hardness magnification* showing that weak lower bounds for some problems imply much stronger lower bounds. Several results of this form have been obtained in different contexts [52, 6, 33, 37, 41], and we refer to [41] for further discussion. Other forms of hardness magnification are known in settings such as communication complexity and arithmetic circuit complexity. A recent example phrased in a way that is closer to our results appears in [11] (see also [18]).

As explained in [6, 41], hardness magnification seems to avoid the natural proofs barrier of [48]. It is therefore important to understand the role of magnification in connection to super-polynomial lower bounds, and this work takes another step in this direction. Our main contributions can be informally described as follows:

- **(i)** We employ new techniques to obtain the first magnification theorem for the worst-case formulation of the MCSP problem.[1]
- **(ii)** Our results establish hardness magnification for a natural meta-computational problem (MKtP) near the lower bound frontiers in several standard circuit models. In addition, we identify a computational model where hardness magnification for MKtP lies below existing lower bounds.
- **(iii)** Crucially, our hardness magnification theorems hold for problems for which it is possible to establish a variety of non-trivial lower bounds.

We believe these results further highlight the relevance of meta-computational problems in connection to the main open problems in algorithms and complexity theory (see e.g. [59, 12] for recent breakthroughs), and strongly indicate that the investigation of weak lower bounds for MKtP and MCSP is a fundamental research direction.

1.2 Results

In this section, we formally state our results. We also briefly discuss some of our techniques, which are explained in more detail in the main body of the paper. We defer a more elaborate discussion of some results to Section 1.3.

Notation. We consider formulas over the bases U_2 (fan-in two ANDs and ORs), B_2 (all boolean functions over two input bits), and extended U_2-formulas where the input leaves are labelled by literals, constants, or parity functions over the input bits of arbitrary arity.

[1] Independently, Dylan McKay, Cody Murray, and Ryan Williams [35] established a magnification theorem for a worst-case formulation of MCSP with a completely different proof.

The corresponding classes of formulas of size at most s (measured by the number of leaves) will be denoted by U_2-Formula$[s]$, B_2-Formula$[s]$, and U_2-Formula-$\oplus[s]$, respectively. If we do not specify the type of formulas, we are referring to De Morgan formulas (i.e., formulas over U_2). We also consider bounded-depth majority circuits, where each internal gate computes a boolean-valued majority function (MAJ) of the form $\sum_{i\in S} y_i \geq^? t$ (the circuit has access to input literals $x_1, \ldots, x_n, \overline{x_1}, \ldots, \overline{x_n}$). We measure the size of such circuits by the number of *wires* in the circuit. Depth-d majority circuits of size s will de denoted by $\mathsf{MAJ}^0_d[s]$, where $d \geq 1$ is fixed. We also consider threshold circuits whose internal gates compute a threshold function (THR) of the form $\sum_{i\in S} w_i \cdot y_i \geq^? t$, for $w_i, t \in \mathbb{R}$. We count number of gates in this case, and let $\mathsf{TC}^0_d[s]$ denote the corresponding class of circuits. Circuit$[s]$ denotes fan-in two boolean circuits of size s and of unbounded depth (gate types do not matter in our results). More generally, for a circuit class $\mathfrak{C}$, we use $\mathfrak{C}[s]$ to denote $\mathfrak{C}$-circuits of size $\leq s$, where size is measured by number of gates. Finally, BP$[s]$ denotes deterministic branching programs of size at most s. We refer to a standard textbook (see e.g. [24]) for more information about these boolean devices.

Gap-MKtP and lower bounds for EXP. We use N to denote the input length of an instance of Gap-MKtP$[s_1, s_2]$ (see Definition 7 below), where we need to distinguish strings of Kt complexity [32] (a certain time-bounded variant of Kolmogorov complexity) at most $s_1(N)$ from strings of Kt complexity at least $s_2(N)$. It is not hard to see that for constructive bounds $s_1 < s_2$, Gap-MKtP$[s_1, s_2] \in$ EXP.

We establish a hardness magnification theorem for Gap-MKtP. (In Section 2, we review some relations between the complexity classes and boolean devices appearing below.) Let $n = \log N$.

▶ **Theorem 1** (Hardness magnification for MKtP). *There is a universal constant $c \geq 1$ for which the following hold. If there exists $\varepsilon > 0$ such that for every small enough $\beta > 0$*

1. Gap-MKtP$[2^{\beta n}, 2^{\beta n} + cn] \notin$ Circuit$[N^{1+\varepsilon}]$, *then* EXP $\not\subseteq$ Circuit[poly].
2. Gap-MKtP$[2^{\beta n}, 2^{\beta n} + cn] \notin U_2$-Formula-$\oplus[N^{1+\varepsilon}]$, *then* EXP $\not\subseteq$ Formula[poly].
3. Gap-MKtP$[2^{\beta n}, 2^{\beta n} + cn] \notin$ AND-THR-THR-XOR$[N^{1+\varepsilon}]$, *then* EXP $\not\subseteq \mathsf{TC}^0_2$[poly].
4. Gap-MKtP$[2^{\beta n}, 2^{\beta n} + cn] \notin \mathsf{MAJ}^0_{2d'+d+1}[N^{1+(2/d')+\varepsilon}]$, *then* EXP $\not\subseteq \mathsf{MAJ}^0_d$[poly].
5. Gap-MKtP$[2^{\beta n}, 2^{\beta n} + cn] \notin B_2$-Formula$[N^{2+\varepsilon}]$, *then* EXP $\not\subseteq$ Formula[poly].
6. Gap-MKtP$[2^{\beta n}, 2^{\beta n} + cn] \notin U_2$-Formula$[N^{3+\varepsilon}]$, *then* EXP $\not\subseteq$ Formula[poly].
7. Gap-MKtP$[2^{\beta n}, 2^{\beta n} + cn] \notin$ BP$[N^{2+\varepsilon}]$, *then* EXP $\not\subseteq$ BP[poly].
8. Gap-MKtP$[2^{\beta n}, 2^{\beta n} + cn] \notin (\mathsf{AC}^0[6])[N^{1+\varepsilon}]$, *then* EXP $\not\subseteq \mathsf{AC}^0[6]$.

Interestingly, this result shows the existence of a single meta-computational problem that is connected to several frontiers in complexity theory.

The proof of Theorem 1 relies on a refinement of some ideas from [41, Section 3.2]. In fact, item 1 of Theorem 1 is a restatement of [41, Theorem 3]. For a sketch of the argument and its underlying techniques, we refer to the discussion in Section 3. We mention that crucial in the proof is the use of error-correcting codes, and that the complexity of computing these objects using different boolean devices gives rise to the distinct magnification thresholds observed in Theorem 1. The formal proof of Theorem 1 appears in Sections 3.1 and 3.2.

In contrast, we observe the following unconditional lower bounds.

▶ **Theorem 2** (Strong lower bounds for large parameters). *For every $\varepsilon > 0$ there exists $\delta > 0$ for which the following results hold:*

1. Gap-MKtP$[2^{(1-\delta)n}, 2^{n-1}] \notin U_2$-Formula$[N^{3-\varepsilon}]$.
2. Gap-MKtP$[2^{(1-\delta)n}, 2^{n-1}] \notin B_2$-Formula$[N^{2-\varepsilon}]$.
3. Gap-MKtP$[2^{(1-\delta)n}, 2^{n-1}] \notin$ BP$[N^{2-\varepsilon}]$.

The proof of Theorem 2 is simple, assuming certain results. It relies on the existence of pseudorandom generators against small formulas and small branching programs [21], together with an observation from [3]. The argument appears in Appendix A.2.

Note the different regime of parameters for $\mathsf{Gap\text{-}MKtP}[s_1, s_2]$ in Theorems 1 and 2. In order to magnify a weak lower bound using Theorem 1, we need that it holds for $s_1 = 2^{o(n)} = N^{o(1)}$. The next result shows that non-trivial unconditional lower bounds can be obtained in this regime.

▶ **Theorem 3** (A near-quadratic formula lower bound)**.** *For every constant $0 < \alpha < 2$ there exists $C > 1$ such that* $\mathsf{Gap\text{-}MKtP}[Cn^2, 2^{(\alpha/2)n-2}] \notin U_2\text{-}\mathsf{Formula}[N^{2-\alpha}]$.[2]

The proof of Theorem 3 adapts ideas from [17, Section 4] (see also the exposition in [41, Appendix C.1]) employed in the context of MCSP for larger parameters. A sketch of the argument followed by a proof can be found in Appendix A.1.

Gap-MCSP and lower bounds for NP. We use $N = 2^n$ to denote the input length of an instance of $\mathsf{Gap\text{-}MCSP}[s_1, s_2]$ (see Definition 9 below), where one needs to distinguish functions of circuit complexity at most s_1 from functions of circuit complexity at least s_2. It is not hard to see that for constructive bounds $s_1 < s_2$, $\mathsf{Gap\text{-}MCSP}[s_1, s_2] \in \mathsf{NP}$.

We establish the following magnification theorem for $\mathsf{Gap\text{-}MCSP}$.

▶ **Theorem 4** (Hardness magnification for MCSP)**.** *There is a universal constant $c \geq 1$ for which the following holds. If there exists $\varepsilon > 0$ such that for every small enough $\beta > 0$*

1. $\mathsf{Gap\text{-}MCSP}[2^{\beta n}/cn, 2^{\beta n}] \notin \mathsf{Circuit}[N^{1+\varepsilon}]$*, then* $\mathsf{NP} \not\subseteq \mathsf{Circuit}[\mathsf{poly}]$.

MCSP and MKtP are quite different problems. In our results, an important distinction is that applying a polynomial-time function to an input of MKtP does not substantially increase its Kt complexity (cf. Proposition 8), but this is not necessarily true in the context of circuit complexity, where the input string represents an entire truth-table. For this reason, the proof of Theorem 4 is completely different from the proof of Theorem 1.

Theorem 4 is our main technical contribution. The argument relies on the notion of *anti-checkers*. Roughly speaking, an anti-checker is a bounded collection $\mathcal{S}$ of inputs associated with a hard function f such that any small circuit C differs from f on some input in $\mathcal{S}$. More precisely, it was established in [34] that any function $f\colon \{0,1\}^n \to \{0,1\}$ that requires circuits of size s admits a collection $\mathcal{S}_f$ containing $O(s)$ strings that is an anti-checker against circuits of size roughly s/n. Our argument makes crucial use of anti-checkers, and en route to Theorem 4 we give a more constructive proof of their existence. (While the proof in [34] uses min-max theory, our proof is combinatorial and self-contained.)

We remark that anti-checkers were first employed for hardness magnification in the context of *proof complexity* [37]. However, while the existential result from [34] was sufficient in that context, this is not the case in *circuit complexity*, and our argument needs to be more sophisticated. For the reader interested in learning more about hardness magnification in proof complexity, how it relates to meta-computational problems such as MCSP, and how the new results compare with previous work, we refer to Appendix B.

The proof of Theorem 4 is not difficult given a certain lemma about the construction of anti-checkers (see Section 4.1). The crucial Anti-Checker Lemma (see Lemma 17) says that $\mathsf{NP} \subseteq \mathsf{Circuit}[\mathsf{poly}]$ implies the existence of circuits of *almost linear size* which given the

[2] The constant C has an exponential dependence on $1/\alpha$.

truth table of a Boolean function f print a corresponding set $\mathcal{S}_f$. The circuits provided by the Anti-Checker Lemma simulate the alternate proof of the existence of anti-checkers, but make the involved argument constructive by using approximate counting and the assumption $\mathsf{NP} \subseteq \mathsf{Circuit}[\mathsf{poly}]$. The strategy for proving the Anti-Checker Lemma is somewhat similar to the proof of $\mathsf{S}_2^p \subseteq \mathsf{ZPP}^{\mathsf{NP}}$ [10]. A high-level exposition and the complete proof are described in Section 4.[3]

▶ Remark. Implicit in our proof of Theorem 4 is a Turing kernelization for the parameterized version of Gap-MCSP which might be of independent interest – there are nearly-linear sized circuits which solve any instance of Gap-MCSP with parameter s using oracle access to $\mathsf{poly}(s)$-sized instances of a fixed language in the Polynomial Hierarchy.

We are able to show the following related unconditional lower bound against *formulas*.

▶ **Theorem 5.** *For each* $0 < \alpha < 2$ *there exists* $d > 1$ *such that* $\mathsf{Gap\text{-}MCSP}[n^d, 2^{(\alpha/2-o(1))n}] \notin U_2\text{-}\mathsf{Formula}[N^{2-\alpha}]$.

Consequently, if one could establish an analogue of Theorem 4 for sub-quadratic formulas, then $\mathsf{NP} \not\subseteq \mathsf{Formula}[\mathsf{poly}]$. We explain why the argument behind the proof of Theorem 4 fails in the case of formulas in Section 4.2.[4] The proof of Theorem 5 is similar to the proof of Theorem 3, and we sketch the necessary modifications in Appendix A.3.

Finally, in Section 4.3 we discuss a certain combinatorial hypothesis ("The Anti-Checker Hypothesis") connected to the techniques behind the proof of Theorem 4. If this hypothesis holds, then $\mathsf{NP} \not\subseteq \mathsf{Formula}[\mathsf{poly}]$. We observe that the hypothesis does hold in the average-case, but we are unsure about its plausibility in the worst-case context that is sufficient for super-polynomial formula lower bounds.

1.3 Discussion

This work is a sequel to an earlier paper of two of the authors [41], in which hardness magnification was first explored in a systematic way. The results in [41] are for a variety of problems (including SAT, Vertex Cover and variants of MKtP and MCSP)[5] and models (including formulas, circuits and sublinear-time algorithms). For each (problem, model) pair considered in [41], it is shown that *non-trivial* lower bounds for the problem against the model imply *super-polynomial* lower bounds for some other explicit problem.

As discussed in [41], there are two natural interpretations of magnification results. The first, more optimistic, interpretation is that magnification constitutes a new approach to proving strong lower bounds. If we are able to replicate the non-trivial circuit lower bounds we can prove against models such as constant-depth circuits (in the worst case) or formulas (in the average case) for the problems witnessing the magnification phenomenon, then this would lead to new and powerful lower bounds. There are no well-understood obstacles to the success of such an approach. In particular, the natural proofs barrier of Razborov and Rudich [48] does not seem to say anything interesting about the success or failure of such an approach.

---

[3] We stress that the assumption that $\mathsf{NP} \subseteq \mathsf{Circuit}[\mathsf{poly}]$ allows several computations to be performed in circuit size $O(N^c)$, where N is the input length. Note however that our requirement is much more stringent: we need to construct anti-checkers using circuits of size $O(N^{1+\varepsilon})$ instead of $O(N^c)$ for some $c \in \mathbb{N}$.

[4] Note that Theorem 4 implies lower bounds for a problem in NP. Theorem 1 only gives lower bounds in EXP, but its proof extends to several low-complexity settings.

[5] The variant of MCSP investigated in [41] is different than the one discussed in this work, and refers to the *average-case* circuit complexity of the input truth table.

The other, more pessimistic, interpretation of magnification results is that they indicate that circuit lower bounds might be *even harder* to achieve than previously thought. Earlier, super-polynomial lower bounds seemed to be out of reach, but there was no strong reasons to believe that small *fixed polynomial* lower bounds or at least *barely non-trivial* lower bounds are hard to show. Modulo the belief that super-polynomial circuit lower bounds for explicit hard problems are hard to show, the magnification phenomenon suggests that for several natural problems of interest, even non-trivial lower bounds are hard to show.

The results of [41] have drawbacks from the point of view of either interpretation, which the present work addresses.

For the optimistic interpretation, it would be good to have examples of natural problems where some magnification phenomenon holds, and where in addition, there are techniques giving non-trivial lower bounds. In this work, we give magnification results for the Gap-MKtP and Gap-MCSP problems, for both of which we show that there are non-trivial lower bounds in the model of Boolean formulas. Thus there is *some* lower bound technique which works to give a non-trivial result – the question is "merely" whether it can be strengthened to derive a lower bound beyond the magnification threshold.

While the pessimistic interpretation might not lead to new lower bounds, it does have the potential of leading to a better understanding of barriers. From this point of view, [41] is not particularly sensitive to the specific model being considered. It is clear that some models are easier to prove lower bounds for than others – indeed we have near-cubic lower bounds in the De Morgan formula model, near-quadratic lower bounds in the branching program model, and only trivial lower bounds in the Boolean circuit model. Can magnification be used to give a new perspective on these differences between models?

We provide a positive answer to this question, by giving different magnification thresholds for different models. What remains mysterious is why known lower bound techniques fall short of proving lower bounds required to apply magnification. This suggests that there are limitations of the known techniques above and beyond those captured by natural proofs – an important direction for further research.

It is worth emphasizing that there are natural problems for which showing lower bounds that are *weaker* than the current state-of-the-art size bounds would also imply super-polynomial lower bounds [41]. A representative example presented in [41] concerns an average-case version of MCSP, where the problem refers to the average-case circuit complexity of the input function. The reason that work does not imply super-polynomial lower bounds via magnification is that the corresponding unconditional lower bounds and magnification theorems hold for a different regime of the average-case complexity parameter.[6]

Our results and techniques were motivated in part by the desire to address this gap. On the one hand, it seems to be easier to analyse problems that refer to the worst-case complexity of the input. But on the other hand, our new results indicate that the shift from average-case to worst-case complexity (in the description of the problem) often increases the magnification threshold to size bounds that are beyond existing techniques. As a concrete example, if the formula magnification theorem for the average-case MCSP problem investigated in [41] could be established for the worst-case variant investigated here, $\mathsf{NP} \not\subseteq \mathsf{NC}^1$ would follow via Theorem 5. Another glimpse of the subtle transition between worst-case and average-case complexity and its role in magnification appears in the discussion of the Anti-Checker Hypothesis in Section 4.3.

[6] In particular, the lower bounds and magnification theorems from [41] do not hold for the same problems.

Complementing these results, we identify a computational model that has not received much attention in the literature, and for which the magnification threshold for Gap-MKtP lies below existing lower bounds. This corresponds to Theorem 4 Item 2, i.e., U_2-formulas augmented with parities in the leaves (our exposition in Section 3 focuses on this model). Note that, by a straightforward simulation, before breaking the cubic barrier for U_2-formulas or the quadratic barrier for B_2-formulas, one needs to show super-linear lower bounds against U_2-Formula-$\oplus$. But a recent result of Tal [55] implies exactly that: the inner product function over N input bits is not in U_2-Formula-$\oplus[N^{1.99}]$.

This makes this computational model particularly attractive in connection to hardness magnification and lower bounds. Indeed, it seems "obvious" that Gap-MKtP$[2^{\delta n}, 2^{\delta n} + cn] \notin U_2$-Formula-$\oplus[N^{1.01}]$, given that such formulas cannot compute the much simpler inner product function, and that standard formulas require at least near-quadratic size (Theorem 3). Our work shows that if this is the case, then $\mathsf{EXP} \not\subseteq \mathsf{NC}^1$.

2 Preliminaries

For $\ell \in \mathbb{N}$, we use $[\ell]$ to denote the set $\{1, \ldots, \ell\}$. The length of a string w will be denoted by $|w|$. Our logarithms are in base 2, and we use $\exp(x)$ to denote e^x. We use boldface symbols such as $\boldsymbol{i}$ and $\boldsymbol{\rho}$ to denote random variables, and $\boldsymbol{x} \in_R S$ to denote that $\boldsymbol{x}$ is a uniformly random element from a set S. We often identify n with $\log N$ or N with 2^n, depending on the context.

For concreteness, we employ a random-access model to formalize uniform algorithms. The details of the model are not crucial in our results, and only mildly affect the gap parameters s_1 and s_2. We fix some standard encoding of algorithms as strings, and use $\langle M \rangle$ to denote the string encoding the algorithm M. Moreover, we assume for simplicity the following property of this encoding: if an algorithm C is obtained via the composition of the computations of algorithms A and B, then $|\langle C \rangle| \leq |\langle A \rangle| + |\langle B \rangle| + O(1)$. (Roughly speaking, composing two codes gives a new valid code.[7]) The running time of M on x is denoted by $t_M(x)$.

We introduce next the notion of Kt complexity. We adopt a formulation that is more convenient for our purposes. In particular, we avoid the use of universal machines in the definition given below.[8] (Our definition is easily seen to be within at most a logarithmic additive term of the formulation using universal machines. We stress that our proofs can be adapted to work with any reasonable definition.)

▶ **Definition 6** (Kt Complexity ([32]; see also [3])). *For a string $x \in \{0,1\}^*$, $\mathsf{Kt}(x)$ denotes the minimum of $|\langle M \rangle| + |a| + \lceil \log t_M(a) \rceil$ over pairs (M, a) such that the machine M outputs x when it computes on the input string a.*

▶ **Definition 7** (The Gap-MKtP Problem). *We consider the promise problem* Gap-MKtP$[s_1, s_2]$, *where $s_1, s_2 \colon \mathbb{N} \to \mathbb{N}$ and $s_1(N) < s_2(N)$ for all $N \in \mathbb{N}$. For each $N \geq 1$,* Gap-MKtP$[s_1, s_2]$

[7] While this holds for instance for programs with relative jump instructions (i.e., goto instructions where the new line is encoded relative to the number of the current line), we remark this is not true in general. For instance, composing two Turing Machines might require renaming all states of one machine, which could result in a new encoding of length $(1 + o(1))|\langle A \rangle| + |\langle B \rangle|$. Depending on the computational model, the results in Theorem 1 might need parameters $s_2 = (1 + o(1))s_1$.

[8] Universal machines are still needed to upper bound the time complexity of computing Kt complexity. Moreover, the exact Kt complexity of a string depends on the choice of encoding for algorithms/machines.

is defined by the following sets of instances:

$$\mathcal{YES}_N \stackrel{\text{def}}{=} \{\, x \in \{0,1\}^N \mid \mathsf{Kt}(x) \leq s_1(N) \,\}, \text{ and}$$
$$\mathcal{NO}_N \stackrel{\text{def}}{=} \{\, x \in \{0,1\}^N \mid \mathsf{Kt}(x) > s_2(N) \,\}.$$

We will need the following simple result.

▶ **Proposition 8** (Kt complexity and composition)**.** *Let B be an algorithm that runs in time at most $T_B(N)$ over inputs of length N. Then, for every input $w \in \{0,1\}^N$, as N grows we have*

$$\mathsf{Kt}(B(w)) \leq \mathsf{Kt}(w) + \log(T_B(N)) + O(1).$$

Proof. Let A be a machine and a be a string such that the pair (A, a) witnesses the value $\mathsf{Kt}(w)$. Let C be the composition of machines A and B, i.e., $C(y) = B(A(y))$. We claim that the pair (C, a) witnesses the inequality in the conclusion of the proposition. Indeed, since $C(a) = B(A(a)) = B(w)$, we get

$$\begin{aligned}
\mathsf{Kt}(B(w)) &\leq |\langle C \rangle| + |a| + \lceil \log t_C(a) \rceil \\
&\leq |\langle A \rangle| + |\langle B \rangle| + O(1) + |a| + \log(t_A(a) + t_B(w)) \\
&\leq |\langle A \rangle| + |a| + \log(t_A(a)) + \log(t_B(w)) + |\langle B \rangle| + O(1) \\
&\leq \mathsf{Kt}(w) + \log(T_B(N)) + O(1),
\end{aligned}$$

where we have used that $|\langle B \rangle|$ is constant as N grows. ◀

We also consider a natural formulation of the gap version of the Minimum Circuit Size Problem (MCSP). The circuit complexity of a boolean function $f\colon \{0,1\}^n \to \{0,1\}$ is denoted by $\mathsf{Size}(f)$. We use the same notation to represent the circuit complexity of the function encoded by a string $x \in \{0,1\}^{2^n}$.

▶ **Definition 9** (The Gap-MCSP Problem)**.** *We consider the promise problem* $\mathsf{Gap\text{-}MCSP}[s_1, s_2]$*, where $s_1, s_2\colon \mathbb{N} \to \mathbb{N}$ and $s_1(n) < s_2(n)$ for all $n \in \mathbb{N}$. For each $n \geq 1$,* $\mathsf{Gap\text{-}MCSP}[s_1(n), s_2(n)]$ *is defined by the following sets of instances:*

$$\mathcal{YES}_n \stackrel{\text{def}}{=} \{\, x \in \{0,1\}^{2^n} \mid \mathsf{Size}(x) \leq s_1(n) \,\}, \text{ and}$$
$$\mathcal{NO}_n \stackrel{\text{def}}{=} \{\, x \in \{0,1\}^{2^n} \mid \mathsf{Size}(x) > s_2(n) \,\}.$$

A brief review of uniform complexity classes and connections to non-uniform devices. To provide some context for Theorem 1, we remind the reader about the following relations involving boolean devices and complexity classes. Under an appropriate uniform formulation of circuit classes, we have the inclusions:

$$\text{(uniform classes)} \quad \mathsf{AC}^0 \subseteq \mathsf{ACC}^0 \subseteq \mathsf{MAJ}^0 \subseteq \mathsf{TC}^0 \subseteq \mathsf{NC}^1 \subseteq \mathsf{L} \subseteq \mathsf{P}.$$

Some of these classes are related in the non-uniform case as follows: $\mathsf{NC}^1 = \mathsf{U}_2\text{-}\mathsf{Formula}[\mathsf{poly}] = \mathsf{B}_2\text{-}\mathsf{Formula}[\mathsf{poly}] =$ (width 5) $\mathsf{BP}[\mathsf{poly}]$, $\mathsf{L/poly} = \mathsf{BP}[\mathsf{poly}]$, $\mathsf{P/poly} = \mathsf{Circuit}[\mathsf{poly}]$, and $\mathsf{MAJ}^0[\mathsf{poly}] = \mathsf{TC}^0[\mathsf{poly}]$. These equivalences might require a complexity overhead in size or depth. We refer to [44, 24] for these and other related results.

3 Hardness Magnification via Error-Correcting Codes

In this section, we prove Theorem 1. First, we provide a high-level exposition of the argument.

Proof Idea. The result is established in the contrapositive. The idea is to reduce Gap-MKtP$[s_1, s_2]$ to a problem in EXP over instances of size $\mathsf{poly}(s_1, s_2) \ll N$, and to invoke the assumed complexity collapse to solve Gap-MKtP using very efficient circuits (or other boolean devices). First, we apply an error-correcting code (ECC) to the input string $w \in \{0,1\}^N$. Since this can be done by a uniform polynomial time computation, we are able to show that $\mathsf{ECC}(w) \in \{0,1\}^{O(N)}$ is a string of Kt complexity $\ell < s_2$ if w has Kt complexity $\leq s_1$. On the other hand, using an efficient decoder for the ECC, we can show that if w has Kt complexity $\geq s_2$, then any string of Kt complexity $> \ell$ differs from $\mathsf{ECC}(w)$ on a constant fraction of coordinates. Let $z = \mathsf{ECC}(w)$. Given the gap in the input instances of Gap-MKtP, our task now is to distinguish strings z that have Kt complexity at most ℓ from strings that cannot be approximated by strings of Kt complexity at most ℓ, where $s_1 < \ell < s_2$.

We achieve this by using a random projection of the input z to a string y of size roughly $\ell \ll N$. The intuition is that if z has Kt complexity at most ℓ, then every projection of z also agrees with some string (i.e., z) of Kt complexity at most ℓ. However, it is possible to argue that if z cannot be approximated by a string of Kt complexity at most ℓ, then with high probability no string of Kt complexity at most ℓ agrees with the randomly projected coordinates of z. Checking which case holds when we are given the string y can be done by an exponential time algorithm. Under the assumption that EXP admits small circuits, we are able to solve this problem in complexity $\mathsf{poly}(\ell) \ll N$.

The reduction sketched above requires (1) the computation of an appropriate ECC, and (2) is randomized. A careful derandomization and the computation of the ECC in different models of computation provide the size bounds corresponding to the magnification thresholds appearing in the statement of Theorem 1.

We start with a detailed proof of Item (2), which covers the more interesting scenario of formulas with parity leaves. We then discuss how a simple modification of the argument together with known results imply the other cases.

3.1 Proof of Theorem 1 Case 2 (Magnification for formulas with parities)

We will need the following explicit construction.

▶ **Theorem 10** (Explicit linear error-correcting codes (cf. [25, 50]))**.** *There exists a sequence $\{E_N\}_{N\in\mathbb{N}}$ of error-correcting codes $E_N\colon \{0,1\}^N \to \{0,1\}^{M(N)}$ with the following properties:*

- *$E_N(x)$ can be computed by a uniform deterministic algorithm running in time $\mathsf{poly}(N)$.*
- *$M(N) = b \cdot N$ for a fixed $b \geq 1$.*
- *There exists a constant $\delta > 0$ such that any codeword $E_N(x) \in \{0,1\}^{M(N)}$ that is corrupted on at most a δ-fraction of coordinates can be uniquely decoded to x by a uniform deterministic algorithm D running in time $\mathsf{poly}(M(N))$.*
- *Each output bit is computed by a parity function: for each input length $N \geq 1$ and for each coordinate $i \in [M(N)]$, there exists a set $S_{N,i} \subseteq [N]$ such that for every $x \in \{0,1\}^N$,*

$$E_N(x)_i = \bigoplus_{j \in S_{N,i}} x_j.$$

We proceed with the proof of Theorem 1 Part (2). We establish the contrapositive. Assume that $\mathsf{EXP} \subseteq \mathsf{Formula}[\mathrm{poly}]$, and recall that $N = 2^n$. For any $\varepsilon > 0$, we prove that $\mathsf{Gap\text{-}MKtP}[2^{\beta n}, 2^{\beta n} + cn] \in U_2\text{-}\mathsf{Formula}\text{-}\oplus[N^{1+\varepsilon}]$ for a sufficiently small $\beta > 0$ and a universal choice of the constant c. The value of c will be specified later in the proof (see Claim 12 below).

Let $E_N \colon \{0,1\}^N \to \{0,1\}^M$ be the error-correcting code granted by Theorem 10, where $M(N) = bN$. Given an instance $w \in \{0,1\}^N$ of $\mathsf{Gap\text{-}MKtP}[2^{\beta n}, 2^{\beta n} + cn]$, we first apply E_N to $w \in \{0,1\}^N$ to get $z = E_N(w) \in \{0,1\}^M$.

▷ Claim 11. There exists $c_0 \geq 1$ such that for every large enough N the following holds. If $\mathsf{Kt}(w) \leq 2^{\beta n}$, then $\mathsf{Kt}(z) \leq 2^{\beta n} + c_0 n$.

Proof. The claim follows immediately from the upper bound on $\mathsf{Kt}(w)$, the definition of $z = E_N(w)$, the running time of E_N, and Proposition 8. ◁

▷ Claim 12. There exist $c > c_1 > c_0 \geq 1$ such that for every large enough N the following holds. If $\mathsf{Kt}(w) > 2^{\beta n} + cn$, then $\mathsf{Kt}(z') > 2^{\beta n} + c_1 n$ for any $z' \in \{0,1\}^M$ that disagrees with z on at most a δ-fraction of coordinates.

Proof. Suppose that a string $z' \in \{0,1\}^M$ disagrees with z on at most a δ-fraction of coordinates, and that $\mathsf{Kt}(z') \leq 2^{\beta n} + c_1 n$ for some $c_1 > c_0$. We upper bound the Kt complexity of w by combining a description of z' with the decoder D provided by Theorem 10. In more detail, assume the pair (F, a) witnesses $\mathsf{Kt}(z')$. Let B be the machine that first applies the machine F to a (producing z'), then D to z'. It follows from Theorem 10 that $B(a) = D(F(a)) = D(z') = w$. Similarly to the proof of Proposition 8, we also get

$$\begin{aligned}
\mathsf{Kt}(w) &\leq |\langle B \rangle| + |a| + \lceil \log t_B(a) \rceil \\
&\leq |\langle F \rangle| + |\langle D \rangle| + O(1) + |a| + \log(t_F(a) + t_D(z')) \\
&\leq \mathsf{Kt}(z') + \log(t_D(z')) + O(1) \\
&\leq (2^{\beta n} + c_1 n) + O(n) + O(1) \\
&\leq 2^{\beta n} + cn,
\end{aligned}$$

if n is large enough and we choose c sufficiently large. ◁

Next we define an auxiliary language $L \in \mathsf{EXP}$, efficiently reduce $\mathsf{Gap\text{-}MKtP}$ to L, and use the assumption that EXP has polynomial size formulas to obtain almost-linear size formulas (of the appropriate kind) for $\mathsf{Gap\text{-}MKtP}$. Roughly speaking, we are able to obtain a formula of non-trivial size for $\mathsf{Gap\text{-}MKtP}$ because our reduction maps input instances of length N to instances of L of length $N^{o(1)}$ (the $o(1)$ term is captured by the parameter β using $n = \log N$). As we will see shortly, the reduction is randomized. In order to get the final U_2-formula-$\oplus$ computing $\mathsf{Gap\text{-}MKtP}$, the argument is derandomized in a straightforward but careful way. More details follow.

An input string y encoding a tuple $(a, 1^b, (i_1, \alpha_1), \ldots, (i_r, \alpha_r))$ belongs to L (where a and b are positive integers, a is encoded in binary, and $\alpha_j \in \{0,1\}$) if each i_j (for $1 \leq j \leq r$) is a string of length $\lceil \log a \rceil$ and there is a string z of length a such that $\mathsf{Kt}(z) \leq b$ and for each index j we have $z_{i_j} = \alpha_j$.

▷ Claim 13. $L \in \mathsf{EXP}$.

Proof. L is decidable in exponential time as we can exhaustively search all strings of Kt complexity at most b and length exactly a and check if there is one which has the specified values at the corresponding bit positions. Indeed, using the definition of Kt complexity and

an efficient universal machine, a list containing all such strings can be generated in time $\mathsf{poly}(2^b)$, which is at most exponential in the input length 1^b. In turn, checking that a string of length a satisfies the requirement takes time at most exponential in the total input length, since each index i_j is a string of length $\lceil \log a \rceil$. ◁

Since $\mathsf{EXP} \subseteq \mathsf{Formula}[\mathsf{poly}]$ by assumption, L has polynomial-size formulas. Assume without loss of generality that L has formulas of size $O(\ell^k)$ for some constant k, where ℓ is its total input length. We choose $\beta = \varepsilon/100k$.

We are ready to describe a low-complexity reduction from $\mathsf{Gap\text{-}MKtP}[2^{\beta n}, 2^{\beta n} + cn]$ to L. First, we use the error-correcting code to compute z from w, as described above. Then we apply the following sampling procedure. We sample uniformly and independently $r = 2^{2\beta n}$ indices $\boldsymbol{i_1}, \ldots, \boldsymbol{i_r} \in_R [M]$, where $M = bN$. We then form the string y encoding the tuple

$$(M, 1^{2^{\beta n} + c_1 n}, (\boldsymbol{i_1}, z_{\boldsymbol{i_1}}), \ldots, (\boldsymbol{i_r}, z_{\boldsymbol{i_r}})),$$

where $c_1 > c_0 \geq 1$ is provided by Claim 12. Note that this is a string of length $\ell(N) \leq N^{\varepsilon/10k}$.

▷ Claim 14. The following implications hold:

(a) If $w \in \{0,1\}^N$ is a positive instance of $\mathsf{Gap\text{-}MKtP}[2^{\beta n}, 2^{\beta n} + cn]$, then $y \in L$ with probability 1.

(b) If $w \in \{0,1\}^N$ is a negative instance of $\mathsf{Gap\text{-}MKtP}[2^{\beta n}, 2^{\beta n} + cn]$, then $y \notin L$ with probability $> 1/2$.

Proof. If w is a YES instance, we have by Claim 11 that $\mathsf{Kt}(z) \leq 2^{\beta n} + c_0 n \leq 2^{\beta n} + c_1 n$. In this case, z is a string of length M that has the specified values at the specified bit positions, regardless of the random positions that are sampled by the reduction. Consequently, $y \in L$ with probability 1.

For the claim about NO instances, as previously established in Claim 12, we have that $\mathsf{Kt}(z') > 2^{\beta n} + c_1 n$ for any z' such that $|z'| = |z| = M$ and $\Pr_{\boldsymbol{i} \in_R [M]}[z'_{\boldsymbol{i}} \neq z_{\boldsymbol{i}}] \leq \delta$. Now consider any string z'' of length M such that $\mathsf{Kt}(z'') \leq 2^{\beta n} + c_1 n$. For such a string z'', for each $j \in [r]$, the probability that the random projection satisfies $z''_{\boldsymbol{i_j}} = z_{\boldsymbol{i_j}}$ (where $\boldsymbol{i_j} \in_R [M]$) is at most $1 - \delta$. Hence the probability that z'' agrees with z at all the specified bit positions is at most $(1-\delta)^r \leq \exp(-\delta r) \leq \exp(-\delta 2^{2\beta n})$. By a union bound over all strings z'' with $\mathsf{Kt}(z'') \leq 2^{\beta n} + c_1 n$, the probability that there exists a string z'' with Kt complexity at most $2^{\beta n} + c_1 n$ which is consistent with the values at the specified bit positions is exponentially small in n. Hence with high probability $y \notin L$. ◁

To sum up, there is a randomized reduction from $\mathsf{Gap\text{-}MKtP}[2^{\beta n}, 2^{\beta n} + cn]$ over inputs of length N to instances of L of length $\ell(N) \leq N^{\varepsilon/10k}$. Now let $\{F_{\ell(N)}\}_{N \geq 1}$ be a sequence of U_2-formulas of size $O(\ell^k)$ for L. Our randomized formulas $\boldsymbol{G}(\cdot)$ for $\mathsf{Gap\text{-}MKtP}$ compute as follows.

1. $\boldsymbol{G}(w) = \bigwedge_{j=1}^{N} \boldsymbol{G}^{(j)}(w)$, where each $\boldsymbol{G}^{(j)}$ is an independent copy.
2. Each $\boldsymbol{G}^{(j)}(w)$ is a randomized formula of the form $G^{(j)}(w, \boldsymbol{i_1}, \ldots, \boldsymbol{i_r})$ that first computes z from w, then computes y from z using the (random) input indices $\boldsymbol{i_1}, \ldots, \boldsymbol{i_r} \in \{0,1\}^{\log M}$, and finally applies F_ℓ to y.

It follows from Claim 14 using the independence of each $\boldsymbol{G}^{(j)}$ that

$$\Pr[\, \boldsymbol{G}(w) \text{ is incorrect} \,] \;<\; 2^{-N},$$

where the probability is taken over the choice of the random input of G. Consequently, by a union bound there is a fixed choice $\gamma \in \{0,1\}^*$ of the randomness of G (corresponding to the positions of the different random projections) such that the *deterministic* formula G_γ obtained from G and γ is correct on *every* input string w.

▷ Claim 15. Each deterministic sub-formula $G_\gamma^{(j)}(w)$ can be computed by a U_2-formula extended with parities at the leaves of size at most $O(\ell(N)^k) \leq N^{\varepsilon/2}$.

Proof. Note that each bit of z can be computed from the input string w using an appropriate parity function (as described in Theorem 10). We argue that the leaves of $G_\gamma^{(j)}$ are precisely the leaves of the U_2-formula F_ℓ replaced by appropriate literals, constants, or parities. Recall that $G_\gamma^{(j)}$ applies F_ℓ to the string y obtained from z. However, since γ is fixed, the positions of z that are projected in order to compute y are also fixed, and so are the substrings of y describing the corresponding positions. Consequently, the size (i.e. number of leaves) of each $G_\gamma^{(j)}$ is at most the size of F_ℓ, which proves the claim. ◁

It follows from this claim that $G_\gamma(w)$ can be computed by a formula containing at most $N^{1+\varepsilon}$ leaves, and hence $\mathsf{Gap\text{-}MKtP}[2^{\beta n}, 2^{\beta n} + cn] \in U_2\text{-}\mathsf{Formula}\text{-}\oplus[N^{1+\varepsilon}]$. (Observe that we have used in a crucial way that the derandomized sub-formulas do not need to compute address functions to generate y from z.) This completes the proof of Theorem 1 Part (2).

3.2 Completing the proof of Theorem 1

In this section, we discuss how the argument presented in Section 3.1 can be adapted to establish the remaining items of Theorem 1.

First, note that Items (5) and (6) immediately follow from Item (2). This is because a parity gate over at most N input variables can be computed by B_2-formulas of size $O(N)$ and by U_2-formulas of size $O(N^2)$. Consequently, using that formula size is measured with respect to the number of leaves, we immediately get $U_2\text{-}\mathsf{Formula}\text{-}\oplus[s(N)] \subseteq B_2\text{-}\mathsf{Formula}[s(N) \cdot N]$ and $U_2\text{-}\mathsf{Formula}\text{-}\oplus[s(N)] \subseteq U_2\text{-}\mathsf{Formula}[s(N) \cdot N^2]$.

In order to get Item (1), it is sufficient to compute an error-correcting code as in Theorem 10 using circuits of (almost) linear size. In other words, we need the entire codeword (and not just each output bit) to be computable from the input message using a circuit of size $O(N)$. The existence of such codes is well-known [50, 51]. The rest of the reduction produces an *additive* overhead in circuit size of at most $N^{1+\varepsilon}$ gates.

Finally, to establish Item (4), we use the following construction from [56].

▶ **Theorem 16** (Computing ECCs in parallel using majorities and few wires [56]). *For every depth $d' \geq 1$ there are constants $\delta(d') > 0$ and $b(d') \geq 1$ and a sequence $\{E_N\}_{N\in\mathbb{N}}$ of error-correcting codes $E_N\colon \{0,1\}^N \to \{0,1\}^M$ with the following properties:*

- *$E_N(x)$ can be computed by a uniform deterministic algorithm running in time $\mathsf{poly}(N)$.*
- *$M(N) = b \cdot N$.*
- *Any codeword $E_N(x) \in \{0,1\}^M$ that is corrupted on at most a δ-fraction of coordinates can be uniquely decoded to x by a uniform deterministic algorithm D running in time $\mathsf{poly}(M)$.*
- *$E_N(x) \in \{0,1\}^M$ can be computed by a multi-output circuit from $\mathsf{MAJ}^0_{2d'}[O(N^{1+(2/d')})]$, where circuit size is measured by number of wires.*

Following the steps of the reduction described in Section 3.1, under the assumption that $\mathsf{EXP} \subseteq \mathsf{MAJ}^0_d[\mathsf{poly}]$ the final depth of the circuit solving $\mathsf{Gap\text{-}MKtP}$ is $2d' + d + 1$, where the terms in this sum correspond respectively to the computation of the error-correcting code (for a choice of $d' \geq 1$), each (circuit) $G_\alpha^{(j)}$, and the topmost AND gate in G_α (constant bits can be produced in depth 1 from input literals). Similarly, the overall size (number of wires) of the circuit is $O(N^{1+(2/d')}) + O(N^{1+\varepsilon}) + O(N) \leq N^{1+(2/d')+\varepsilon}$.

Item (3) is established in the obvious way given the previous explanations. Item (8) uses that parity gates can be simulated using mod 6 gates.

Finally, we deal with case (7), which refers to branching program complexity. First, note that the parity of n bits can be computed by a branching program of size $O(n)$. In addition, if $f(x) = g(h_1(x), \ldots, h_k(x))$, each h_i has a branching program of size s, and g has a branching program of size t, then f has a branching program of size $\ell = O(t \cdot s)$. Finally, a conjunction of N branching programs of size ℓ has branching program size at most $O(N \cdot \ell)$. Combining these facts in the natural way yields case (7). This completes the proof of all cases in Theorem 1.

4 Hardness Magnification via Anti-Checkers

4.1 Proof of Theorem 4 (Magnification for MCSP)

In this section, we derive Theorem 4 from Lemma 17, whose proof appears in Section 4.2. Informally, an anti-checker (cf. [34]) for a function f is a multi-set of input strings such that any circuit of bounded size that does not compute f is incorrect on at least one of these strings.

▶ **Lemma 17** (Anti-Checker Lemma). *If* $\mathsf{NP} \subseteq \mathsf{Circuit}[\mathsf{poly}]$ *there is a constant* $k \in \mathbb{N}$ *for which the following hold. For every sufficiently small* $\beta > 0$*, there is a circuit* C *of size* $\leq 2^{n+k\beta n}$ *that when given as input a truth-table* $\mathsf{tt}(f) \in \{0,1\}^N$*, where* $f\colon \{0,1\}^n \to \{0,1\}$*, outputs* $t = 2^{10\beta n}$ *strings* $y_1, \ldots, y_t \in \{0,1\}^n$ *such that if* $f \notin \mathsf{Circuit}[2^{\beta n}]$ *then every circuit of size* $\leq s$ *where* $s = 2^{\beta n}/10n$ *fails to compute* f *on at least one of these strings.*

The Anti-Checker Lemma is a powerful tool that might be of independent interest. It says that anti-checkers of bounded size for functions requiring circuits of size $2^{o(n)}$ can be produced in time that is almost-linear in the size of the function (viewed as a string), under the assumption that circuit lower bounds do not hold.[9]

Proof of Theorem 4. Assume that $\mathsf{NP} \subseteq \mathsf{Circuit}[\mathsf{poly}]$. We prove that for every given $\varepsilon > 0$ there exists a small enough $\beta > 0$ such that $\mathsf{Gap\text{-}MCSP}[2^{\beta n}/10n, 2^{\beta n}] \in \mathsf{Circuit}[N^{1+\varepsilon}]$.

We consider the problem $\mathsf{Succinct\text{-}MCSP}$, defined next. Its input instances are of the form $\langle 1^n, 1^s, 1^t, (x_1, b_1), \ldots, (x_t, b_t)\rangle$, where $x_i \in \{0,1\}^n$ and $b_i \in \{0,1\}$, $i \in [t]$. Note that each instance can be encoded by a string of length exactly $m = n + 1 + s + 1 + t + 1 + t \cdot (n+1)$. An input string is a positive instance if and only if it is in the appropriate format and there exists a circuit D over n input variables and of size at most s such that $D(x_i) = b_i$ for all $i \in [t]$. Note that the problem is in NP as a function of its total input length m. Under the assumption that NP is easy for non-uniform circuits, there exists $\ell \in \mathbb{N}$ such that $\mathsf{Succinct\text{-}MCSP}$ can be solved by circuits E_m of size m^ℓ on every large enough input length m.

Take $\beta = \varepsilon/(100 \cdot \ell \cdot k)$, where k is the constant from Lemma 17. In order to construct a circuit for $\mathsf{Gap\text{-}MCSP}$, first we reduce this problem to an instance of $\mathsf{Succinct\text{-}MCSP}$ of length m using Lemma 17, then we invoke the m^l-sized circuit for this problem. More precisely, on an input $f\colon \{0,1\}^n \to \{0,1\}$, we use the circuit C (as in Lemma 17) to produce a list of strings $y_1, \ldots, y_t \in \{0,1\}^n$, generate from this list and f the input instance $z = \langle 1^n, 1^s, 1^t, ((y_1, f(y_1)), \ldots, (y_t, f(y_t))\rangle$, for parameters $s = 2^{\beta n}/10n$, $t = 2^{10\beta n}$, $m = \mathsf{poly}(n) \cdot 2^{10\beta n}$, and output $E_m(z)$.

[9] We have made no attempt to optimize the constants in Lemma 17.

Correctness follows immediately from Lemma 17 and our choice of parameters. Indeed, if $f \in \mathsf{Circuit}[2^{\beta n}/10n]$ then no matter the choice of $y_1, \ldots, y_t$ the circuit E_m accepts z thanks to our choice of $s = 2^{\beta n}/10n$. On the other hand, when $f \notin \mathsf{Circuit}[2^{\beta n}]$ then by Lemma 17 every circuit of size s fails on some string from the list, and consequently $E_m(z) = 0$.

We upper bound the total circuit size using the choice of β. Circuit C has size at most $2^{n+k\beta n} \leq N^{1+\varepsilon}/3$. In addition, producing the input z can be done from f and $y_1, \ldots, y_t$ by a circuit of size at most $O(t \cdot N) \leq N^{1+\varepsilon}/3$, since each address function can be computed in linear size $O(N)$ (see e.g. [58]). Finally, E_m has size at most $m^\ell \leq N^{1+\varepsilon}/3$. Overall, it follows that $\mathsf{Gap\text{-}MCSP}[2^{\beta n}/10n, 2^{\beta n}]$ is computable by circuits of size $N^{1+\varepsilon}$. ◀

4.2 Proof of Lemma 17 (Anti-Checker Lemma)

This section is dedicated to the proof of Lemma 17. This completes the proof of Theorem 4. We start with a high-level exposition of the argument.

Proof Idea. We take $\beta \to 0$, for simplicity of the exposition. In principle, the challenge is to *construct* the list of strings from the description of f using a circuit of size $N^{1+o(1)}$, given that the *existence* of such strings is guaranteed by the work of [34]. But it is not clear how to use this existential result and the assumption that NP has polynomial size circuits to construct *almost-linear* size circuits for this task. In order to achieve this, we use a *self-contained* argument that produces the strings one by one until very few circuits of bounded size are consistent with the values of f on the partial list of strings. We then find polynomially many *additional* strings that eliminate the remaining circuits, completing the list of strings.[10]

To produce the i-th string $y_i \in \{0,1\}^n$ given $y_1, \ldots, y_{i-1} \in \{0,1\}^n$ and f, we estimate the number of circuits of size $\leq 2^{\beta n}/10n$ that agree with f over all strings in $\{y_1, \ldots, y_i\}$. We show that *some string* y_i will reduce the number of consistent circuits from the previous round by a factor of (roughly) $1 - 1/n$ if there are at least (roughly) n^2 surviving circuits (this is a combinatorial existential proof that relies on the lower bound on the circuit complexity of f). As a consequence, it will be possible to show that at most $2^{O(\beta n)} = N^{o(1)}$ rounds suffice to produce the required set of strings (modulo handling the few surviving circuits). The existence of a *good* string y_i is at the heart of our argument, and we defer the exposition of this result to the formal proof.

In each round, we exhaustively check each of the N candidate strings y_i. As we will explain soon, estimating the number of surviving circuits after picking a new candidate string y_i can be done by a circuit of size $N^{o(1)}$ given access to $y_1, \ldots, y_i$ and to the corresponding bits $f(y_1), \ldots, f(y_i)$.[11] In summary, there are $N^{o(1)}$ rounds, and in each one of them we can find a good string y_i using a circuit of size $N^{1+o(1)}$. We remark that it will also be possible to produce the additional strings in circuit complexity $N^{o(1)}$, so that the complete list $y_1, \ldots, y_t$ can be computed from f by a circuit of size $N^{1+o(1)}$.

It remains to explain how to fix a good string in each round. We simply pick the most promising string, using that we can upper bound the complexity of estimating the number of surviving circuits. The latter relies on the assumed inclusion $\mathsf{NP} \subseteq \mathsf{Circuit}[\mathsf{poly}]$. Indeed, from

[10] In particular, our argument implies the worst-case version of the anti-checker result from [34] with slightly different parameters.

[11] Technically speaking, projecting $f(y_i) \in \{0,1\}$ from the input string $f \in \{0,1\}^N$ and the address $y \in \{0,1\}^n$ already takes circuit complexity $\Omega(N)$. However, since we are trying all possible strings y_i, the corresponding bit positions of f can be directly hardwired.

this assumption it follows that the polynomial hierarchy $\mathsf{PH} \subseteq \mathsf{Circuit}[\mathsf{poly}]$, and it is known that *relative approximate counting* can be done in the polynomial hierarchy.[12] Crucially, as described in the paragraph above, the input length of each sub-problem that we need to solve is $\leq N^{o(1)}$ (using that i is at most $N^{o(1)}$), so a polynomial overhead will not be an issue when solving a sub-task of input length $N^{o(1)}$. This completes the sketch of the proof.

We proceed with a formal proof of Lemma 17. Let R be a polynomial-time relation, where $R \subseteq \bigcup_m \{0,1\}^m \times \{0,1\}^{q(m)}$ for some polynomial q. For every x, we use $R_\#(x)$ to denote $|\{y \in \{0,1\}^{q(|x|)} : (x,y) \in R\}|$. A randomized algorithm $\mathbf{\Pi}$ is called an (ε,δ)-*approximator* for R if for every input x it holds that

$$\Pr\big[\,\big|\mathbf{\Pi}(x) - R_\#(x)\big| \;\geq\; \varepsilon(|x|)\cdot R_\#(x)\,\big] \;\leq\; \delta(|x|).$$

▶ **Theorem 18** (Relative approximate counting in $\mathsf{BPP}^{\mathsf{NP}}$ ([53]; see e.g. [15, Section 6.2.2]))**.** *For every polynomial-time relation R and every polynomial p, there exists a probabilistic polynomial-time algorithm $\boldsymbol{A}$ with access to a SAT oracle that is an $(1/p(m), 2^{-p(m)})$-approximator for R over inputs x of length m.*

▶ **Corollary 19.** *Assume $\mathsf{NP} \subseteq \mathsf{Circuit}[\mathsf{poly}]$. For every polynomial-time relation R and for each $m \geq 1$, there is a multi-output circuit $C_R : \{0,1\}^m \to \{0,1\}^{\mathsf{poly}(m)}$ of polynomial size such that on every input $x \in \{0,1\}^m$,*

$$(1 - 1/m^2)\cdot R_\#(x) \;\leq\; C_R(x) \;\leq\; (1 + 1/m^2)\cdot R_\#(x).$$

Proof. This follows from Theorem 18 (using $p(m) = m^2$) by non-uniformly fixing the randomness of the algorithm, replacing the SAT oracle using the assumption that NP has small circuits, and translating the resulting deterministic algorithm into a boolean circuit. ◀

We define a relation Q. The first input x is of the form $\langle 1^n, 1^s, 1^i, 1^t, (z_1,b_1), \ldots, (z_i,b_i)\rangle$, where $z_j \in \{0,1\}^n$ and $b_j \in \{0,1\}$ for $1 \leq j \leq i$, and $t = 2^{10\beta n}$ (t is used here to pad the input appropriately). The second input is a string w of length $m^{1/5}$ (for $m = |x|$) that is interpreted as a boolean circuit C_w over n input variables and of size at most s. We let $(x,w) \in Q$ if and only if $C_w(z_j) = b_j$ for all $j \in [i]$. Note that Q is a polynomial-time relation.

We employ circuits obtained from Corollary 19 using parameters $s = 2^{\beta n}/10n$ and $1 \leq i \leq t$, where $t = 2^{10\beta n}$. The following result is immediate from Corollary 19 given that for our choice of parameters $m = \mathsf{poly}(2^{\beta n})$.

▶ **Proposition 20** (Circuits for approximate counting)**.** *There is a constant $k_1 \in \mathbb{N}$ for which the following holds. For every $n \geq 1$, let $s = 2^{\beta n}/10n$, $t = 2^{10\beta n}$, $1 \leq i \leq t$. Then there is a multi-output circuit $C_{n,i}$ of size $\leq 2^{k_1\beta n}$ that outputs $\leq 2^{k_1\beta n}$ bits such that on every input $a = ((z_1,b_1),\ldots,(z_i,b_i)) \in \{0,1\}^{i\cdot(n+1)}$,*

$$(1 - 1/n^{10})\cdot Q_\#(x) \;\leq\; C_{n,i}(a) \;\leq\; (1 + 1/n^{10})\cdot Q_\#(x),$$

where $x = x(a)$ is defined from the string a and from our choice of parameters in the obvious way.

The next step is to guarantee that once just a few circuits remain consistent with f over our partial list of strings (as described in the proof sketch above), we can efficiently find a small number of strings to eliminate all of them.

[12] In our formal proof, we take a slightly more direct route to compute the relative approximations.

▶ **Lemma 21** (Listing the remaining circuits). *Assume* NP $\subseteq$ Circuit[poly]. *There exists a constant* $k_2 \in \mathbb{N}$ *for which the following holds. Let* $a = ((z_1, b_1), \ldots, (z_{t'}, b_{t'}))$, *where* $t' \leq t$, *and* $x = x(a)$ *be the corresponding input of* Q. *There is a circuit* $D_{n,t'}$ *of size* $\leq 2^{k_2 \beta n}$ *such that if* $Q_{\#}(x) \leq n^3$, *then* $D_{n,t'}(a)$ *outputs a string describing all such circuits.*

Proof. It follows from NP $\subseteq$ Circuit[poly] using a standard argument that PH $\subseteq$ Circuit[poly]. In addition, it is not hard to define a relation in PH (using a padded input containing the string 1^t) that checks if a given input a satisfies $Q_{\#}(x(a)) \leq n^3$. Consequently, checking if a string λ describes a list of such circuits for a can be done by a circuit of size at most $\mathsf{poly}(t)$. Using again that NP $\subseteq$ Circuit[poly] and a self-reduction, we obtain circuits $D_{n,t'}$ as in the statement of the lemma. ◀

▶ **Lemma 22** (Completing the list of strings). *There is a constant* $k_3 \in \mathbb{N}$ *for which the following holds. For every* $n \geq 1$ *there is a circuit* E_n *of size* $\leq 2^{n+k_3\beta n}$ *that given access to a truth-table* $f \in \{0,1\}^{2^n}$ *and a string* $w \in \{0,1\}^{2^{\beta n}}$ *describing a circuit* C_w *of size* $s \leq 2^{\beta n}/10n$ *that does not compute* f, $E_n(f, w)$ *outputs a string* y *such that* $C(y) \neq f(y)$.

Proof. First, E_n evaluates C_w on every string $z \in \{0,1\}^n$. This can be easily done by a circuit of size $2^n \cdot \mathsf{poly}(|w|)$ under a reasonable encoding of the circuit C_w. Then E_n inspects one-by-one each string z and stores the first string where C_w and f differ. Note that a circuit of size $\leq 2^n \cdot \mathsf{poly}(n)$ can print this string from the truth-table of f and C_w. It follows that the overall complexity of E_n is $2^{n+k_3\beta n}$ for some constant k_3. ◀

The previously established results will allow us to find in each round a string y_i that significantly reduces the number of remaining circuits (while at least one such string exists), and then to complete the list so that no circuit of bounded size is consistent with all strings in the final list. We show next that if f is hard and a reasonable number of circuits of bounded size are consistent with the current list of strings, then a good string y_i exists.

For convenience, we introduce a function to capture the fraction of strings encoding circuits that are consistent with a set of inputs and their corresponding labels. Given $a = ((z_1, b_1), \ldots, (z_i, b_i))$, let $x = x(a)$ be the corresponding input to Q under our choice of parameters. Furthermore, let $m = |x|$, and recall that $Q \subseteq \bigcup_{m \geq 1} \{0,1\}^m \times \{0,1\}^{m^{1/5}}$. In order to maintain the same underlying domain size when considering the fraction of consistent circuits, we assume without loss of generality using appropriate padding that the encoding of x has a fixed length $m = m(n)$ for each choice of n (i.e., the choice of $1 \leq i \leq t$ does not affect $m^{1/5}$). In addition, we can take $m(n) \leq 2^{11\beta n}$, which will be useful when upper bounding the number of necessary rounds. We let $\phi(a) \in [0,1]$ denote the ratio $Q_{\#}(x(a))/2^{m^{1/5}}$. (Thus in our formal argument we count circuits using their descriptions as binary strings.)

▶ **Lemma 23** (Existence of a good string y_i). *For every integer* $i \geq 1$ *and for every* $z_1, \ldots, z_{i-1} \in \{0,1\}^n$, *let* $a = ((z_1, f(z_1)), \ldots, (z_{i-1}, f(z_{i-1})))$. *If*

$$f \notin \mathsf{Circuit}[2^{\beta n}] \quad \textit{and} \quad Q_{\#}(x(a)) \geq 4n^2,$$

then there is some string $y_i \in \{0,1\}^n$ *such that if* a' *denotes the sequence* a *augmented with* $(y_i, f(y_i))$, *then*

$$\phi(a') \;\leq\; \phi(a) \cdot (1 - 1/2n).$$

Proof. The argument is inspired by a combinatorial principle discussed in [29]. Consider the tuple a and the string $x = x(a)$ as in the statement of the lemma. Moreover, let $Q(x) = \{w \in \{0,1\}^{m^{1/5}} : (x, w) \in Q\}$. For convenience, let $r = |Q(x)| = Q_{\#}(x) \geq 4n^2$, using

our assumption. Define an auxiliary undirected bipartite graph $G = (L, R, E)$ as follows. Set $L = \{0,1\}^n$, $R = \binom{Q(x)}{n}$, and $(y, \{w^1, \ldots, w^n\}) \in E(G)$ if and only if for $\leq n/2$ of the circuits C_{w^i} we have $f(y) = C_{w^i}(y)$.

Note that for any right vertex $v = (w^1, \ldots, w^n) \in R$ there is a left vertex $y \in L$ such that $(y, v) \in E$. If not, then $D = \mathsf{Majority}_n(C_{w^1}(x), \ldots, C_{w^n}(x))$ is a circuit that computes f on every input string y. The size of D is at most $n \cdot (2^{\beta n}/10n) + 5n \leq 2^{\beta n}$, using the definition of Q and that the majority function can be computed (with room to spare) by a circuit of size at most $5n$ [58]. This contradicts the hardness of f.

By an averaging argument, there is a left vertex y^* that is connected to at least $|R|/|L| = \binom{r}{n}/2^n$ vertices in R. We show below (Claim 24) that for at least $r/2n$ strings $w \in Q(x)$, the corresponding circuit C_w satisfies $C_w(y^*) \neq f(w^*)$. This implies that by taking y^* as the string y_i described in the statement of the lemma, we get $Q_{\#}(x(a')) \leq r - r/2n = r(1-1/2n)$, and consequently

$$\phi(a') \;=\; \frac{Q_{\#}(x(a'))}{2^{m^{1/5}}} \;\leq\; \frac{r(1-1/2n)}{2^{m^{1/5}}} \;=\; \frac{Q_{\#}(x(a)) \cdot (1-1/2n)}{2^{m^{1/5}}} \;=\; \phi(a) \cdot (1-1/2n).$$

▷ Claim 24. Let $y^* \in L$ be a left-vertex connected to at least $\binom{r}{n} \cdot 2^{-n}$ right-vertices in R, where $r \geq 4n^2$ and n is sufficiently large. Then, for at least $r/2n$ distinct strings $w \in Q(x)$, we have $C_w(y^*) \neq f(y^*)$.

Proof. The claim follows using a standard counting argument. If the conclusion were false, the vertex y^* would be connected to *strictly* less than (assuming for simplicity that n is even and $r/2n$ is an integer)

$$\sum_{j=0}^{n/2} \binom{r/2n}{\frac{n}{2}+j} \cdot \binom{r}{\frac{n}{2}-j} \;\leq\; \binom{r}{n} \cdot 2^{-n} \quad \text{(as upper bounded below)}$$

vertices in R, which is contradictory. It remains to verify this inequality, which can be done using some careful estimates. First, note that

$$\begin{aligned}
\sum_{j=0}^{n/2} \binom{r/2n}{\frac{n}{2}+j}\binom{r}{\frac{n}{2}-j} &\leq \sum_{j=0,\ldots,\frac{n}{2}-1} \frac{r^n/(2n)^{\frac{n}{2}+j}}{(\frac{n}{2}+j)!(\frac{n}{2}-j)!} + \frac{r^n}{n!(2n)^n} && \left(\text{using } \binom{n}{k} \leq \frac{n^k}{k!}\right)\\
&\leq \sum_{j=0,\ldots,\frac{n}{2}-1} \frac{e^n r^n/(2n)^{\frac{n}{2}+j}}{e^2(\frac{n}{2}+j)^{\frac{n}{2}+j}(\frac{n}{2}-j)^{\frac{n}{2}-j}} + \frac{e^n r^n}{en^n(2n)^n} && \left(\text{since } e\left(\frac{n}{e}\right)^n \leq n!\right)\\
&\leq \sum_{j=0,\ldots,\frac{n}{2}-1} \frac{e^n r^n/(2n)^{\frac{n}{2}}}{e^2(\frac{n}{2})^j(\frac{n}{2}+j)^{\frac{n}{2}}(\frac{n}{2}-j)^{\frac{n}{2}}} + \frac{e^n r^n}{en^n(2n)^n} && (*)
\end{aligned}$$

By considering the cases $j < \frac{n}{4}$ and $\frac{n}{2} > j \geq \frac{n}{4}$, we get $(\frac{n}{2})^j((\frac{n}{2})^2 - j^2)^{\frac{n}{2}} \geq (n/8)^{3n/4}$, so

$$\begin{aligned}
(*) \;&\leq \sum_{j=0,\ldots,\frac{n}{2}-1} \frac{e^n r^n}{e^2(n/8)^{3n/4}(2n)^{n/2}} + \frac{e^n r^n}{en^n(2n)^n}\\
&\leq \frac{ne^n r^n}{e^2(n/8)^{3n/4}(2n)^{n/2}} \;\leq\; \frac{\sqrt{2\pi}r^n}{e^2 n^{1/2}(2n)^n} \;\leq\; \frac{\sqrt{2\pi}r^r r^{1/2}}{e^2(r-n)^{r-n+1/2}n^{n+1/2}} \cdot \frac{1}{2^n}\\
&\leq \binom{r}{n}/2^n,
\end{aligned}$$

where n is assumed to be sufficiently large, $r > n$, and the last inequality makes use of Stirling's approximation $\sqrt{2\pi}(\frac{n}{e})^n n^{1/2} \leq n! \leq e(\frac{n}{e})^n n^{1/2}$. This completes the proof of Claim 24. ◁

This completes the proof of Lemma 23. ◀

We are ready to combine these results and define a circuit C of size $\leq 2^{n+k\beta n}$ with the property stated in Lemma 17. This circuit on an input $f \in \{0,1\}^N$ where $N = 2^n$ computes as follows.

1. C sequentially computes the string $a^{(i)} = (y_1, f(y_1)), \ldots, (y_i, f(y_i))$ for $1 \leq i \leq t'$ and $t' = 2^{10\beta n} - n^3$.

During stage i, C inspects all strings $y \in \{0,1\}^n$, using the circuit $C_{n,i}$ (Proposition 20) to fix y_i as the string that minimizes $C_{n,i}(a^{(i)})$.

2. C uses the circuit $D_{n,t'}$ (Lemma 21) to print the descriptions of n^3 circuits of size at most $s = 2^{\beta n}/10n$.

3. Finally, C invokes n^3 copies of the circuit E_n (Lemma 22) to complete the list $y_1, \ldots, y_t$ of strings, where $t = t' + n^3 = 2^{10\beta n}$.

Correctness of the construction follows from the properties of the circuits $C_{n,i}$, $D_{n,t'}$, and E_n in combination with Lemma 23. More precisely, if $f \notin \mathsf{Circuit}[2^{\beta n}]$, then for every $1 \leq i \leq t'$, either $\phi(a^{(i)}) \leq (1 - 1/4n)^i$ or $Q_{\#}(x(a^{(i-1)})) < 4n^2$. To see this, note that if the latter condition does not hold, then for some string y^* as in Lemma 23 we get with respect to the corresponding extension $a^{(i)}$ that $\phi(a^{(i)}) \leq \phi(a^{i-1}) \cdot (1 - 1/2n)$. Since C tries all strings during its computation in step **1** when in stage i, and the relative approximation given by circuit $C_{n,i}$ is sufficiently precise, we are guaranteed in this case (using an inductive argument) to fix a string y_i such that $\phi(a^{(i)}) \leq \phi(a^{i-1}) \cdot (1 - 1/4n) \leq (1 - 1/4n)^i$. On the other hand, if the condition $Q_{\#}(x(a^{(i-1)})) < 4n^2$ holds for some $i \leq t'$, then by monotonicity it is maintained until we reach $i = t'$. Consequently, using that initially $\phi(\epsilon) = 1$, $t' = 2^{10\beta n} - n^3$, $m(n) \leq 2^{11\beta n}$, and recalling that the second input of the relation Q has length $m^{1/5}$ and that this parameter is related to the definition of ϕ, when C reaches $i = t'$ at the end of step **1** we have

$$\begin{aligned} Q_{\#}(x(a^{(t')})) &\leq \max\{4n^2, (1 - 1/4n)^{t'} \cdot 2^{m^{1/5}}\} \\ &\leq n^3. \end{aligned}$$

This implies using Lemmas 21 and 22 and the description of C that if $f \notin \mathsf{Circuit}[2^{\beta n}]$ then every circuit of size at most $s = 2^{\beta n}/10n$ disagrees with f on some input string among $y_1, \ldots, y_t$.

Finally, we upper bound the circuit size of C. For every $i \leq t'$ in step **1** and each string $y \in \{0,1\}^n$, C feeds $C_{n,i}$ with the appropriate bit in the input string f and the previously computed string $a^{(i-1)}$. This produces an estimate $v_y \in \mathbb{N}$ represented as a string of length $2^{O(\beta n)}$ that is stored as a pair (y, v_i). Using Proposition 20, all pairs (y, v_y) can be simultaneously computed by a circuit of size at most $2^n \cdot 2^{O(\beta n)}$. By inspecting each such pair in sequence, C can pick the string $y_i \in \{0,1\}^n$ minimizing v_i using a sub-circuit of size $2^n \cdot \mathsf{poly}(2^{O(\beta n)})$. Also note that the bit $f(y_i)$ can be easily computed from y_i and f by a circuit of size $O(N \log N)$. Therefore, each stage i can be done by a circuit of size at most $2^{n+O(\beta n)}$, and since there are $t' \leq 2^{10\beta n}$ stages, the computation in step **1.** can be done by a circuit of size $2^{n+O(\beta n)}$. Lastly, steps **2** and **3** can be each implemented by a circuit of size at most $2^{O(\beta n)}$ using the upper bounds on circuit size provided by Lemmas 21 and 22, respectively, and the description of C. It follows that the overall circuit size of C is at most $2^{n+k\beta n}$, where k is a constant that only depends on the circuits provided by the initial assumption that $\mathsf{NP} \subseteq \mathsf{Circuit}[\mathsf{poly}]$.

A remark on formulas vs. circuits. An obstacle to producing the anti-checker using formulas of size $N^{1+o(1)}$ under the assumption that $\mathsf{NP} \subseteq \mathsf{Formula}[\mathsf{poly}]$ comes from the sequential aspect of the construction. A string y_j produced after the j-th round is inspected during each subsequent round of the construction. In the case of formulas, the corresponding bits need to be recomputed each time, and the overall complexity becomes prohibitive. (There are other intermediate computations that one may not be able to simulate so easily with sub-quadratic formulas, such as selecting the best string y_i during each round.)

4.3 The Anti-Checker Hypothesis

The existence of anti-checkers of bounded size witnessing the hardness of Boolean functions is far from obvious. In this section, we explore consequences of a hypothetical phenomenon manifesting on a higher level: the existence of a small collection of anti-checker sets witnessing hardness of all hard functions. We show that a certain formulation of this Anti-Checker Hypothesis (AH) implies unconditional lower bounds. Complementing this result, we prove unconditionally that (AH) holds for functions that are hard in the average case.

For simplicity, we adopt a concrete setting of parameters for the hypothesis and in the results presented in this section. Understanding the validity of (AH) with respect to other non-trivial setting of parameters would also be interesting.

▶ **The Anti-Checker Hypothesis (AH).** *For every $\lambda \in (0,1)$, there are $\varepsilon > 0$ and a collection $\mathcal{Y} = \{Y_1, \ldots, Y_\ell\}$ of sets $Y_i \subseteq \{0,1\}^n$, where $\ell = 2^{(2-\varepsilon)n}$ and each $|Y_i| = 2^{n^{1-\epsilon}}$, for which the following holds.*

If $f\colon \{0,1\}^n \to \{0,1\}$ and $f \notin \mathsf{Circuit}[2^{n^\lambda}]$, then some set $Y \in \mathcal{Y}$ forms an anti-checker for f: For each circuit C of size $2^{n^\lambda}/10n$, there is an input $y \in Y$ such that $C(y) \neq f(y)$.

The Anti-Checker Hypothesis can be shown to imply the hardness of a specific meta-computational problem in NP (which is not necessarily NP-complete).

▶ **Definition 25** (Succinct MCSP). *Let $s, t\colon \mathbb{N} \to \mathbb{N}$ be functions. The Succinct Minimum Circuit Size Problem with parameters s and t, abbreviated $\mathit{Succinct\text{-}MCSP}(s,t)$, is the problem of deciding given a list of $t(n)$ pairs (y_i, b_i), where $y_i \in \{0,1\}^n$ and $b_i \in \{0,1\}$, if there exists a circuit C of size $s(n)$ computing the partial function defined by these pairs, i.e., $C(y_i) = b_i$ for every $i \in [t]$.*

Note that $\mathsf{Succinct\text{-}MCSP}(s,t) \in \mathsf{NP}$ whenever s and t are constructive functions.

▶ **Theorem 26.** *Assume* (AH) *holds, and let $\varepsilon = \varepsilon(\lambda) > 0$ be the corresponding constant for $\lambda = 1/2$. Then $\mathit{Succinct\text{-}MCSP}(2^{n^{1/2}}/10n, 2^{n^{1-\varepsilon}}) \notin \mathsf{Formula}[\mathsf{poly}]$. In particular, $\mathsf{NP} \not\subseteq \mathsf{Formula}[\mathsf{poly}]$.*

Proof. The proof is by contradiction. Take $\lambda = 1/2$ in the Anti-Checker Hypothesis, and let $\varepsilon = \varepsilon(\lambda) > 0$ be the given constant. In addition, let $F_m\colon \{0,1\}^N \to \{0,1\}$ be a formula of size m^k for $\mathsf{Succinct\text{-}MCSP}(2^{n^{1/2}}/10n, 2^{n^{1-\varepsilon}})$, where $m \leq \mathsf{poly}(n) \cdot 2^{n^{1-\varepsilon}}$ is the total input length for this problem. We argue below that from these assumptions it follows that $\mathsf{Gap\text{-}MCSP}[2^{n^{1/3}}, 2^{n^{2/3}}] \in \mathsf{Formula}[N^{2-\delta}]$ for some $\delta > 0$. This contradicts Theorem 5 if α is taken to be a sufficiently small constant, which completes the proof.

We define a formula $E\colon \{0,1\}^N \to \{0,1\}$ that solves $\mathsf{Gap\text{-}MCSP}[2^{n^{1/3}}, 2^{n^{2/3}}]$. It projects the appropriate bits of the input f to produce $T = 2^{(2-\varepsilon)n}$ instances of the problem $\mathsf{Succinct\text{-}MCSP}(2^{n^{1/2}}/10n, 2^{n^{1-\varepsilon}})$ obtained from f and from the collection $\mathcal{Y}$ in the natural way. The formula E is defined as the conjunction of T independent copies of the formula F_m from above. Note that E has at most $T \cdot m^k \leq N^{2-\delta}$ leaves, where $\delta = \delta(\varepsilon) > 0$. Finally, it is easy to see that it correctly solves $\mathsf{Gap\text{-}MCSP}$ using our choice of parameters and (AH). ◀

We say that a Boolean function f with n inputs is hard on average for circuits of size s if every circuit of size s fails to compute f on at least $1/s$ fraction of all inputs.

▶ **Proposition 27** (Average-Case AH). *For every $\lambda \in (0,1)$ there is $\varepsilon > 0$ such that for every large enough $n \in \mathbb{N}$ there is a collection $\mathcal{Y} = \{Y_1, \ldots, Y_\ell\}$ of $\ell = 2^n$ sets $Y_i \subseteq \{0,1\}^n$ of size $2^{n^{1-\epsilon}}$ for which the following holds. If $f\colon \{0,1\}^n \to \{0,1\}$ is hard on average for circuits of size 2^{n^λ}, then some set $Y \in \mathcal{Y}$ constitutes an anti-checker for f: For each circuit C of size 2^{n^λ} there is a string $y \in Y$ such that $C(y) \neq f(y)$.*

Proof. Let $\mathcal{H}$ be the set of all Boolean functions f over n inputs that are hard on average for circuits of size $s = 2^{n^\lambda}$. Then we can generate anti-checkers for $f \in \mathcal{H}$ by choosing n-bit strings uniformly at random: for each $i \in [2^n]$, we let $\boldsymbol{Y_i}$ be the set obtained by sampling with repetition $2^{n^{1-\varepsilon}}$ random strings in $\{0,1\}^n$, where $1 - \varepsilon > \lambda$. Then, for every large enough n, for each circuit C of size at most 2^{n^λ} and for each $f \in \mathcal{H}$,

$$\Pr[C|_{\boldsymbol{Y_i}} \equiv f|_{\boldsymbol{Y_i}}] \;\leq\; (1 - 1/2^{n^\lambda})^{2^{n^{1-\epsilon}}} \;\leq\; \exp(-2^{n^{1-\epsilon}}/2^{n^\lambda}).$$

Now by a union bound over all such circuits, for a fixed $f \in \mathcal{H}$ we get

$$\Pr[\boldsymbol{Y_i} \text{ is not an anti-checker set for } f] \;\leq\; \exp(O(n \cdot 2^{n^\lambda})) \cdot \exp(-2^{n^{1-\epsilon}}/2^{n^\lambda}) \;<\; 1/4,$$

where the last inequality used our choice of ε. Finally,

$$\Pr[\exists f \in \mathcal{H} \text{ s.t. none of } \boldsymbol{Y_1}, \ldots, \boldsymbol{Y_{2^n}} \text{ is an anti-checker set for } f] \;\leq\; 2^{2^n} \cdot (1/4)^{2^n} \;<\; 1.$$

There is therefore a collection $\mathcal{Y}$ with the desired properties. ◀

Theorem 26 and Proposition 27 show a connection between establishing super-polynomial formula size lower bounds for NP and understanding the difference between worst-case and average-case collections of anti-checkers.

References

1. Scott Aaronson. P $=^?$ NP. *Electronic Colloquium on Computational Complexity* (ECCC), 24:4, 2017. URL: `https://eccc.weizmann.ac.il/report/2017/004`.
2. Michael Alekhnovich, Eli Ben-Sasson, Alexander A. Razborov, and Avi Wigderson. Pseudorandom generators in propositional proof complexity. *SIAM Journal of Computing*, 34(1):67–88, 2004.
3. Eric Allender. When Worlds Collide: Derandomization, Lower Bounds, and Kolmogorov Complexity. In *Foundations of Software Technology and Theoretical Computer Science* FSTTCS, pages 1–15, 2001. `doi:10.1007/3-540-45294-X_1`.
4. Eric Allender. The Complexity of Complexity. In *Computability and Complexity - Essays Dedicated to Rodney G. Downey on the Occasion of His 60th Birthday*, pages 79–94, 2017.
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.
6. Eric Allender and Michal Koucký. Amplifying lower bounds by means of self-reducibility. *J. ACM*, 57(3):14:1–14:36, 2010.
7. Andrej Bogdanov. Small-bias require large formulas. In *International Colloquium on Automata, Languages, and Programming* (ICALP), 2018.
8. Ravi B. Boppana and Michael Sipser. The Complexity of Finite Functions. In *Handbook of Theoretical Computer Science, Volume A: Algorithms and Complexity*, pages 757–804. The MIT Press/Elsevier, 1990.

9 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.

10 Jin-yi Cai. S_2^p is subset of $\mathsf{ZPP}^{\mathsf{NP}}$. *J. Comput. Syst. Sci.*, 73(1):25–35, 2007. doi:10.1016/j.jcss.2003.07.015.

11 Marco Carmosino, Russell Impagliazzo, Shachar Lovett, and Ivan Mihajlin. Hardness Amplification for Non-Commutative Arithmetic Circuits. In *Computational Complexity Conference* (CCC), 2018.

12 Marco L. Carmosino, Russell Impagliazzo, Valentine Kabanets, and Antonina Kolokolova. Learning Algorithms from Natural Proofs. In *Conference on Computational Complexity* (CCC), pages 10:1–10:24, 2016. doi:10.4230/LIPIcs.CCC.2016.10.

13 Irit Dinur and Or Meir. Toward the KRW Composition Conjecture: Cubic Formula Lower Bounds via Communication Complexity. In *Conference on Computational Complexity* (CCC), pages 3:1–3:51, 2016. doi:10.4230/LIPIcs.CCC.2016.3.

14 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 *Symposium on Foundations of Computer Science* (FOCS), pages 89–98, 2016.

15 Oded Goldreich. *Computational Complexity - A Conceptual Perspective*. Cambridge University Press, 2008.

16 Johan Håstad. The Shrinkage Exponent of de Morgan Formulas is 2. *SIAM J. Comput.*, 27(1):48–64, 1998. doi:10.1137/S0097539794261556.

17 Shuichi Hirahara and Rahul Santhanam. On the Average-Case Complexity of MCSP and Its Variants. In *Computational Complexity Conference* (CCC), pages 7:1–7:20, 2017.

18 Pavel Hrubeš, Avi Wigderson, and Amir Yehudayoff. Non-commutative circuits and the sum-of-squares problem. *Journal of the American Mathematical Society*, 24(3):871–898, 2011.

19 Russell Impagliazzo, Valentine Kabanets, and Ilya Volkovich. The power of natural properties as oracles. In *Computational Complexity Conference* (CCC), 2018.

20 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.

21 Russell Impagliazzo, Raghu Meka, and David Zuckerman. Pseudorandomness from Shrinkage. In *Symposium on Foundations of Computer Science* (FOCS), pages 111–119, 2012.

22 Russell Impagliazzo, Ramamohan Paturi, and Michael E. Saks. Size-Depth Tradeoffs for Threshold Circuits. *SIAM J. Comput.*, 26(3):693–707, 1997. doi:10.1137/S0097539792282965.

23 Kazuo Iwama and Hiroki Morizumi. An Explicit Lower Bound of $5n - o(n)$ for Boolean Circuits. In *Symposium on Mathematical Foundations of Computer Science* (MFCS), pages 353–364, 2002.

24 Stasys Jukna. *Boolean Function Complexity - Advances and Frontiers*. Springer, 2012.

25 Jørn Justesen. Class of constructive asymptotically good algebraic codes. *IEEE Trans. Information Theory*, 18(5):652–656, 1972. doi:10.1109/TIT.1972.1054893.

26 Valentine Kabanets and Jin-yi Cai. Circuit minimization problem. In *Symposium on Theory of Computing* (STOC), pages 73–79, 2000. doi:10.1145/335305.335314.

27 Daniel M. Kane and Ryan Williams. Super-linear gate and super-quadratic wire lower bounds for depth-two and depth-three threshold circuits. In *Symposium on Theory of Computing* (STOC), pages 633–643, 2016. doi:10.1145/2897518.2897636.

28 Ilan Komargodski, Ran Raz, and Avishay Tal. Improved Average-Case Lower Bounds for De Morgan Formula Size: Matching Worst-Case Lower Bound. *SIAM J. Comput.*, 46(1):37–57, 2017. doi:10.1137/15M1048045.

29 Jan Krajíček. Extensions of models of PV. In *ASL/Springer Series – Lecture Notes in Logic – Proceedings of the Logic Colloquium*, 1995.

30 Jan Krajíček. On the weak pigeonhole principle. *Fundamenta Mathematicae*, 170(1-3):123–140, 2001.

31 Jan Krajíček. Dual weak pigeonhole principle, pseudo-surjective functions, and provability of circuit lower bounds. *Journal of Symbolic Logic*, 69(1):265–286, 2004.

32 Leonid Levin. Randomness Conservation Inequalities; Information and Independence in Mathematical Theories. *Information and Control*, 61:15–37, 1984.

33 Richard J. Lipton and Ryan Williams. Amplifying circuit lower bounds against polynomial time, with applications. *Computational Complexity*, 22(2):311–343, 2013.

34 Richard J. Lipton and Neal E. Young. Simple strategies for large zero-sum games with applications to complexity theory. In *Symposium on Theory of Computing* (STOC), pages 734–740, 1994. doi:10.1145/195058.195447.

35 Dylan M. McKay, Cody D. Murray, and R. Ryan Williams. Weak Lower Bounds on Resource-Bounded Compression Imply Strong Separations of Complexity Classes. In *Symposium on Theory of Computing* (STOC), 2019.

36 Eric Miles and Emanuele Viola. Substitution-Permutation Networks, Pseudorandom Functions, and Natural Proofs. *J. ACM*, 62(6):46:1–46:29, 2015.

37 Moritz Müller and Ján Pich. Feasibly constructive proofs of succinct weak circuit lower bounds. *Electronic Colloquium on Computational Complexity* (ECCC), 24:144, 2017.

38 Cody Murray and R. Ryan Williams. Circuit lower bounds for nondeterministic quasi-polytime: an easy witness lemma for NP and NQP. In *Symposium on Theory of Computing* (STOC), pages 890–901, 2018. doi:10.1145/3188745.3188910.

39 È. Nečiporuk. On a boolean function. *Doklady of the Academy of the USSR*, 169(4):765–766, 1966.

40 Igor Carboni 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.

41 Igor Carboni Oliveira and Rahul Santhanam. Hardness Magnification for Natural Problems. In *Symposium on Foundations of Computer Science* (FOCS), pages 65–76, 2018. doi:10.1109/FOCS.2018.00016.

42 Ján Pich. *Complexity Theory in Feasible Mathematics*. PhD thesis, Charles University in Prague, 2014.

43 Ján Pich. Circuit lower bounds in bounded arithmetics. *Annals of Pure and Applied Logic*, 166(1), 2015.

44 Alexander A. Razborov. Lower Bounds for Deterministic and Nondeterministic Branching Programs. In *Symposium on Fundamentals of Computation Theory* (FCT), pages 47–60, 1991.

45 Alexander A. Razborov. On provably disjoint NP-pairs. *Basic Research in Computer Science Center*, 1994.

46 Alexander A. Razborov. Unprovability of lower bounds on circuit size in certain fragments of bounded arithmetic. *Izvestiya of Russian Academy of Science,*, 59:201–224, 1995.

47 Alexander A. Razborov. Pseudorandom generators hard for k-DNF resolution and polynomial calculus. *Annals of Mathematics*, 182(2):415–472, 2015.

48 Alexander A. Razborov and Steven Rudich. Natural Proofs. *J. Comput. Syst. Sci.*, 55(1):24–35, 1997.

49 Rahul Santhanam. Circuit Lower Bounds for Merlin–Arthur Classes. *SIAM J. Comput.*, 39(3):1038–1061, 2009. doi:10.1137/070702680.

50 Michael Sipser and Daniel A. Spielman. Expander codes. *IEEE Trans. Information Theory*, 42(6):1710–1722, 1996. doi:10.1109/18.556667.

51 Daniel A. Spielman. Linear-time encodable and decodable error-correcting codes. *IEEE Trans. Information Theory*, 42(6):1723–1731, 1996. doi:10.1109/18.556668.

52 Aravind Srinivasan. On the approximability of clique and related maximization problems. *J. Comput. Syst. Sci.*, 67(3):633–651, 2003.

53 Larry J. Stockmeyer. The Complexity of Approximate Counting (Preliminary Version). In *Symposium on Theory of Computing* (STOC), pages 118–126, 1983.

54 Avishay Tal. Shrinkage of De Morgan Formulae by Spectral Techniques. In *Symposium on Foundations of Computer Science* (FOCS), pages 551–560, 2014.

55 Avishay Tal. The Bipartite Formula Complexity of Inner-Product is Quadratic. *Electronic Colloquium on Computational Complexity* (ECCC), 23:181, 2016.

56 Roei Tell. Quantified Derandomization of Linear Threshold Circuits. In *Symposium on Theory of Computing* (STOC), 2018.

57 Salil P. Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 7(1-3):1–336, 2012.

58 Ingo Wegener. *The complexity of Boolean functions*. Wiley, 1987.

59 Ryan Williams. Nonuniform ACC Circuit Lower Bounds. *J. ACM*, 61(1):2:1–2:32, 2014. doi:10.1145/2559903.

60 Ryan Williams. Natural Proofs versus Derandomization. *SIAM J. Comput.*, 45(2):497–529, 2016. doi:10.1137/130938219.

A Unconditional Lower Bounds for Gap-MKtP and Gap-MCSP

A.1 MKtP – A near-quadratic lower bound against U_2-formulas

In this section, we provide the proof of Theorem 3.

Proof Idea. We employ the technique of random restrictions to show that Gap-MKtP requires near-quadratic size formulas. The idea is that, with high probability, a formula F of sub-quadratic size simplifies under a random restriction $\boldsymbol{\rho}\colon [N] \to \{0,1,*\}$. This will allow us to complete a fixed restriction ρ either to a string w^y of Kt complexity $\leq s_1$, or to a string w^n of Kt complexity $\geq s_2$. Because the simplified formula $F\restriction_\rho$ depends on few input variables in $\rho^{-1}(*)$, if we define w^y and w^n appropriately $F\restriction_\rho$ won't be able to distinguish the two instances. Consequently, F does not compute Gap-MKtP$[s_1, s_2]$.

In order for this idea to work, we cannot use a truly random restriction. This is because our restrictions will set most of the variables indexed in $[N]$ to simplify a near-quadratic size formula, and a typical random restriction cannot be completed to a string of low Kt complexity. We use instead pseudorandom restrictions, which can be computed from a much smaller number of random bits. Previous work established that such restrictions also simplify sub-quadratic size formulas. As a consequence, we are able to extend any restriction in the support of a pseudorandom distribution of restrictions to either an "easy" or a "hard" string, as explained in the paragraph above. (We remark that in order to improve our parameter s_1 in Gap-MKtP$[s_1, s_2]$, it is useful to compose a sequence of pseudodeterministic restrictions.)

We proceed with the technical details. Let $\rho\colon [N] \to \{0,1,*\}$ be a *restriction*, and $\boldsymbol{\rho}$ be a *random restriction*, i.e., a distribution of restrictions. We say that $\boldsymbol{\rho}$ is *p-regular* if $\Pr[\boldsymbol{\rho}(i) = *] = p$ and $\Pr[\boldsymbol{\rho}(i) = 0] = \Pr[\boldsymbol{\rho}(i) = 1] = (1-p)/2$ for every $i \in [N]$. In addition, $\boldsymbol{\rho}$ is *k-wise independent* if any k coordinates of $\boldsymbol{\rho}$ are independent.

▶ **Lemma 28** (cf. [21, 57]). *There exist q-regular k-wise independent random restrictions $\boldsymbol{\rho}$ distributed over $\rho\colon [N] \to \{0,1,*\}$ samplable with $O(k \log(N) \log(1/q))$ bits. Furthermore, each output coordinate of the random restriction can be computed in time polynomial in the number of random bits.*

As a consequence, we get p-regular k-wise independent random restrictions where each restriction in the support has bounded Kt complexity. In order to define the Kt complexity of

a restriction $\rho\colon [N] \to \{0,1,*\}$, we view it as a $2N$-bit string $\mathsf{encoding}(\rho)$ where each symbol in $\{0,1,*\}$ is encoded by an element in $\{0,1\}^2$. We abuse notation and write $\mathsf{Kt}(\rho)$ to denote $\mathsf{Kt}(\mathsf{encoding}(\rho))$.

▶ **Proposition 29.** *There is a distribution $\mathcal{D}_{q,k}$ of q-regular k-wise independent restrictions such that each restriction $\rho\colon [N] \to \{0,1,*\}$ in the support of $\mathcal{D}_{q,k}$ satisfies $\mathsf{Kt}(\rho) = O(k \log(N) \log(1/q))$. Furthermore, this is witnessed by a pair (M, w_ρ) where the machine M does not depend on ρ.*

Proof. By Lemma 28, each output coordinate of ρ can be computed in time $\mathsf{poly}(\ell)$ from a seed w_ρ of length $\ell = O(k \log(N) \log(1/q))$. Therefore, the binary string describing ρ can be computed in time $O(N \cdot \mathsf{poly}(\ell))$ from a string w_ρ with $\mathsf{Kt}(w_\rho) = O(k \log(N) \log(1/q))$. It follows from Proposition 8 that $\mathsf{Kt}(\rho) = O(k \log(N) \log(1/q))$. The furthermore part follows from the fact that the machine M is obtained from the generator provided by Lemma 28, i.e., in order to produce different restrictions one only needs to modify the input seeds, which are encoded in w_ρ. ◀

Let $N = 2^n$. Given a function $F\colon \{0,1\}^N \to \{0,1\}$ and a restriction $\rho\colon [N] \to \{0,1,*\}$, we let $F\restriction_\rho$ be the function in $\{0,1\}^{\rho^{-1}(*)} \to \{0,1\}$ obtained in the natural way from F and ρ. In this section, we use $L(F)$ to denote the size (number of leaves) of the smallest U_2-formula that computes a function F.

The next result allows us to shrink the size of a formula using a pseudorandom restriction. This restriction can be obtained by a composition of restrictions. This reduces the amount of randomness and the corresponding complexity of the restriction.

▶ **Lemma 30** (Shrinkage from pseudorandom restrictions ([17, Theorem 28]; cf. [21, 28]))**.** *Let $F\colon \{0,1\}^N \to \{0,1\}$, $q = p^{1/r}$ for an integer $r \geq 1$, and $L(F) \cdot p^2 \geq 1$. Moreover, let $\mathcal{R}^r_{p,k}$ be a distribution obtained by the composition of r independent q-regular k-wise independent random restrictions supported over $[N] \to \{0,1,*\}$, where $k = q^{-2}$. Finally, assume that $q \leq 10^{-3}$. Then,*[13]

$$\mathbb{E}_{\boldsymbol{\rho} \in_R \mathcal{R}^r_{p,k}}[L(F\restriction_{\boldsymbol{\rho}})] \;\leq\; c^r p^2 L(F),$$

where $c \geq 1$ is an absolute constant.

▶ **Proposition 31.** *There is a (p-regular k-wise independent) distribution $\mathcal{R}^r_{p,k}$ obtained by the composition of r independent q-regular k-wise independent random restrictions supported over $[N] \to \{0,1,*\}$, where $k = q^{-2}$ and $q = p^{1/r}$, such that each restriction $\rho\colon [N] \to \{0,1,*\}$ in the support of $\mathcal{R}^r_{p,k}$ satisfies $\mathsf{Kt}(\rho) = O(rk \log(N) \log(1/q))$.*

Proof. We use the distribution $\mathcal{D}_{q,k}$ of restrictions provided by Proposition 29. A restriction ρ in the support of $\mathcal{R}^r_{p,k}$ is therefore obtained through the composition of r restrictions $\rho_1, \ldots, \rho_r$ in the support of $\mathcal{D}_{q,k}$. For each $i \in [r]$, $\mathsf{Kt}(\rho_r) = O(k \log(N) \log(1/q))$. Moreover, each Kt

[13] The assumption that $q \leq 10^{-3}$ does not appear in [17, Theorem 28]. The proof sketch appearing there does not seem to address the cases where $p^\Gamma L(\psi) < 1$ in their analyses of formula shrinkage in Lemma 27 and Theorem 28. This can be easily fixed using appropriate expressions of the form $1 + p^2 L(\psi)$. Lemma 27 is only affected by a constant factor. Then, proceeding by induction as in the proof of their Theorem 28 but also addressing this possibility, one gets instead an upper bound of the form $1 + cq^\Gamma(1 + cq^\Gamma(\ldots))$, which translates to $1 + (cq^\Gamma) + (cq^\Gamma)^2 + \ldots + (cq^\Gamma)^{r-1} + (cq^\Gamma)^r L(f)$. This can still be upper bounded by $c^r p^2 L(F)$ (for a different universal constant c as in the statement of Lemma 30) using that q is sufficiently small and therefore $cq^\Gamma \leq 1/2$ (note that $\Gamma = 2$ and $c \leq 500$ in [17]).

upper bound is witnessed by a pair (M, w_i), where M can be taken to be the same machine for all $i \in [r]$. It is not hard to see that for the string $w = 1^{|w_1|}0w_1 1^{|w_2|}0w_2 \dots 1^{|w_r|}0w_r$ there is a machine M' satisfying $|\langle M' \rangle| \leq |\langle M \rangle| + O(1)$ and running in time $t_{M'}(w) \leq r \cdot \max_i t_M(w_i) + \mathsf{poly}(rN)$ such that the pair (M', w) witnesses that $\mathsf{Kt}(\rho) = O(rk \log(N) \log(1/q))$. ◀

We will also need the following simple proposition, which holds even with respect to Kolmogorov complexity instead of Kt complexity.

▶ **Proposition 32.** *Let $S \subseteq [N]$ be a set of size at least two. There exists a function $h\colon S \to \{0,1\}$ such that for every string $w \in \{0,1\}^N$, if w agrees with h over S then $\mathsf{Kt}(w) \geq |S| - 5\log|S|$.*

Proof. It is easy to encode a pair (M, a) (as in Definition 6) satisfying $|\langle M \rangle| + |a| < |S| - 5\log|S|$ by a binary string of length at most $2\log|S| + 2 + |\langle M \rangle| + |a| < |S|$. Since each pair (M, a) outputs at most one binary string of length N, it follows by a counting argument that for some choice of $h\colon S \to \{0,1\}$, no string w of length N that agrees with h over S has $\mathsf{Kt}(w) < |S| - 5\log|S|$. ◀

The next lemma describes the high-level strategy of the lower bound proof.

▶ **Lemma 33** (Adaptation of Lemma 27 from [17])**.** *There exists a constant $a \geq 1$ such that the following holds. Let $\rho\colon [N] \to \{0,1,*\}$ be a restriction, $V = \rho^{-1}(*)$, and let $F\colon \{0,1\}^N \to \{0,1\}$ be a function such that $L(F\restriction_\rho) \leq M$. If*

$$\mathsf{Kt}(\rho) + a \cdot n \;\leq\; s_1(n) \quad \textit{and} \quad (|V| - M) - 5\log(|V| - M) \;\geq\; s_2(n) \quad \textit{and} \quad |V| \geq M + a,$$

then F does not compute $\mathsf{Gap\text{-}MKtP}[s_1(n), s_2(n)]$, where $n = \log N$.

Proof. Under these assumptions, we define a positive instance $w^y \in \mathcal{YES}_N$ and a negative instance $w^n \in \mathcal{NO}_N$ such that $F(w^y) = F(w^n)$.

- $w^y \in \{0,1\}^N$ is obtained from ρ by additionally setting each $*$-coordinate of this restriction to 0. Note that, given the $2N$-bit binary string encoding ρ, w^y can be computed in time polynomial in N. It follows from Proposition 8 that $\mathsf{Kt}(w^y) \leq \mathsf{Kt}(\rho) + a \cdot n$, for some universal constant $a \geq 1$. Since this bound is at most $s_1(n)$, we get that $w^y \in \mathcal{YES}_N$.
- $w^n \in \{0,1\}^N$ is defined as follows. Since $L(F\restriction_\rho) \leq M$, $F\restriction_\rho$ depends on at most M input coordinates (indexed by elements in V). Let $W \subseteq V \subseteq [N]$ be this set of coordinates. Moreover, let $S = V \setminus W$. The string $w^y \in \{0,1\}^N$ is obtained from ρ by additionally setting each $*$-coordinate of this restriction in W to 0, and then setting each remaining $*$-coordinate in S to agree with the function $h\colon S \to \{0,1\}$ provided by Proposition 32. Since $|S| \geq |V| - M$ and the real-valued function $\phi(x) = x - 5\log x$ is non-decreasing if $x \geq a$ for a large enough constant a, our assumptions and Proposition 32 imply that $\mathsf{Kt}(w^n) \geq s_2(n)$. Consequently, $y^n \in \mathcal{NO}_N$.

Using that F restricted to ρ depends only on variables from $W \subseteq \rho^{-1}(*)$, and that the strings w^y and w^n agree over coordinates in $\rho^{-1}(\{0,1\}) \cup W$, it follows that $F(w^y) = F(w^n)$. Since w^y is a positive instance while y^n is a negative instance, F does not compute $\mathsf{Gap\text{-}MKtP}[s_1(n), s_2(n)]$. ◀

We are now ready to set parameters in order to complete the proof of Theorem 3. For a sufficiently large constant $C' \geq 1$, let

$$n \stackrel{\text{def}}{=} \log N, \quad p \stackrel{\text{def}}{=} N^{-1+\alpha/2}, \quad r \stackrel{\text{def}}{=} n/C', \quad q \stackrel{\text{def}}{=} p^{1/r}, \quad k \stackrel{\text{def}}{=} q^{-2},$$

and assume that N is sufficiently large. Note that, under this choice of parameters, $q = 2^{C'(-1+\alpha/2)} = \Omega(1)$ and $q \leq 10^{-3}$.

▶ **Proposition 34** (Concentration Bound for $|\boldsymbol{\rho}^{-1}(*)|$). *For $\boldsymbol{\rho} \sim \mathcal{R}^r_{p,k}$ with parameters as above, we have* $\Pr[\,|\boldsymbol{\rho}^{-1}(*)| \geq pN/2\,] \geq 1/2$.

Proof. Note that $\boldsymbol{\rho}$ is p-regular and pairwise independent (i.e., $k \geq 2$ for our choice of parameters). The result then follows from Chebyshev's inequality using mean $\mu = pN$, variance $\sigma^2 = Np(1-p)$, and the value of p. ◀

Using Proposition 31, we can sample a random restriction $\boldsymbol{\rho} \in_R \mathcal{R}^r_{p,k}$ as described in the statement of Lemma 30 such that each $\rho\colon [N] \to \{0,1,*\}$ in the support of $\mathcal{R}^r_{p,k}$ satisfies

$$\mathsf{Kt}(\rho) \;=\; O(rk\log(N)\log(1/q)) \;=\; O((n/C')q^{-2}n\log(1/q)) \;\leq\; (C/2)n^2,$$

if C is a sufficiently large constant.

Toward a contradiction, let $F\colon \{0,1\}^N \to \{0,1\}$ be a formula of size $L(f)$ that supposedly computes $\mathsf{Gap\text{-}MKtP}[Cn^2, 2^{(\alpha/2)n-2}]$, where $p^2L(F) = 1$ (note that $L(F) = N^{2-\alpha}$), and let

$$M \stackrel{\text{def}}{=} 10 \cdot c^r p^2 L(F) \;=\; 10 \cdot c^r \;\leq\; 2^{(\alpha/4)n},$$

for a constant $c \geq 1$ as in Lemma 30, and using that $C' = C'(\alpha)$ is large enough in the definition of r.

Invoking Lemma 30 and Markov's inequality, Proposition 34, and a union bound, there is a fixed restriction $\rho\colon [N] \to \{0,1,*\}$ for which the following holds:

- For $V \stackrel{\text{def}}{=} \rho^{-1}(*)$, we have $|V| \;\geq\; pN/2 \;=\; 2^{(\alpha/2)n}/2$;
- $\mathsf{Kt}(\rho) \;\leq\; (C/2)n^2$.
- $L(F\restriction_\rho) \;\leq\; M \;\leq\; 2^{(\alpha/4)n}$.

Using these parameters in the statement of Lemma 33, it is easy to check that its hypotheses are satisfied given our choices of $s_1(n) = Cn^2$ and $s_2(n) = 2^{(\alpha/2)n-2}$. This is a contradiction to our assumption that F computes $\mathsf{Gap\text{-}MKtP}$ for these parameters, which completes the proof.

A.2 MKtP – Stronger lower bounds for large parameters

The goal of this section is to prove Theorem 2. First, we need a definition. We say that a generator $G\colon \{0,1\}^r \to \{0,1\}^N$ *δ-fools* a function $f\colon \{0,1\}^N \to \{0,1\}$ if

$$\left| \Pr_{\boldsymbol{x} \in_R \{0,1\}^N}[f(x) = 1] - \Pr_{\boldsymbol{y} \in_R \{0,1\}^r}[f(G(y)) = 1] \right| \;\leq\; \delta.$$

Similarly, G δ-fools a class of functions $\mathcal{F}$ if G δ-fools every function $f \in \mathcal{F}$. The parameter r is called the *seed-length* of G. We say that G is *explicit* if it can be uniformly computed in time $\mathsf{poly}(N, 1/\delta)$.

▶ **Theorem 35** ([21]). *Let $c > 0$ be an arbitrary constant. The following hold:*

1. *There is an explicit generator $G^{U_2}\colon \{0,1\}^r \to \{0,1\}^N$ using a seed of length $r = s^{1/3+o(1)}$ that s^{-c}-fools the class U_2-$\mathsf{Formula}[s(N)]$ of formulas on N input variables.*

2. *There is an explicit generator* $G^{B_2}\colon \{0,1\}^r \to \{0,1\}^N$ *using a seed of length* $r = s^{1/2+o(1)}$ *that* s^{-c}*-fools the class* B_2-$\mathsf{Formula}[s(N)]$ *of formulas on* N *input variables.*
3. *There is an explicit generator* $G^{BP}\colon \{0,1\}^r \to \{0,1\}^N$ *using a seed of length* $r = s^{1/2+o(1)}$ *that* s^{-c}*-fools the class* $\mathsf{BP}[s(N)]$ *of branching programs on* N *input variables.*

We now prove Theorem 2 (Part 1). The other cases are similar. We instantiate G^{U_2} with $s(N) = N^{3-\varepsilon}$ and $c = 1$. Then $G^{U_2}\colon \{0,1\}^{N^{1-\delta'}} \to \{0,1\}^N$ for some $\delta' = \delta'(\varepsilon) > 0$.

▶ **Proposition 36.** *For every string* $w \in \{0,1\}^{N^{1-\delta'}}$*, let* $G^{U_2}(w) \in \{0,1\}^N$ *be the* N*-bit output of* G^{U_2} *on* w*. Then*

$$\mathsf{Kt}(G^{U_2}(w)) \;\leq\; 2^{(1-\delta'/2)n}$$

for every large enough $n = \log N$*.*

Proof. This follows from Proposition 8 using that G^{U_2} is explicit and therefore runs in time $\mathsf{poly}(N)$ under our choice of parameters. ◀

As a consequence of Proposition 36, every output of G^{U_2} is always an N-bit string of Kt complexity at most $2^{(1-\delta)n}$, for a fixed $\delta > 0$. On the other hand, it is well-known that a random N-bit string (where $N = 2^n$) has Kolmogorov complexity (and thus Kt complexity) at least 2^{n-1} with high probability. It follows that $\mathsf{Gap\text{-}MKtP}[2^{(1-\delta)n}, 2^{n-1}] \notin U_2$-$\mathsf{Formula}[N^{3-\varepsilon}]$, since otherwise this would violate the security of the generator G^{U_2} against formulas of this type and size.

A.3 MCSP – A similar near-quadratic lower bound against U_2-formulas

In this section, we sketch the proof of Theorem 5, which is the analogue of Theorem 3 in the context of MCSP. More precisely, we explain why the argument carries over when we measure the complexity of a string by circuit size instead of via Kt complexity, modulo small changes to the involved parameters.

As explained in Section A.1, the crucial idea in the proof of Theorem 3 is that a pseudorandom restriction simplifies a U_2-formula of bounded size. For technical reasons, we employ a composition of restrictions of small complexity, so that the overall complexity of the combined restriction is bounded. This allows us to trivialize any small formula F using a fixed restriction ρ of bounded complexity, where $|\rho^{-1}(*)|$ is sufficiently large compared to other relevant parameters of the argument. Then, Lemma 33 employs a counting argument (via Proposition 32) to extend this restriction to a positive instance w^y and to a negative instance w^n such that $F(w^y) = F(w^n)$. This can be used to show that no small formula correctly computes $\mathsf{Gap\text{-}MKtP}$ for our choice of parameters.

In order to establish Theorem 5, we make two observations. Firstly, Lemma 28 already gives individual restrictions of low *circuit complexity* instead of low Kt complexity. Secondly, the *counting argument* used to extend ρ to a negative instance w^n works for most complexity measures including circuit size, Kolmogorov complexity, etc.

Using these two observations, the proof goes through under minor adjustments of the relevant parameters. We remark that one obtains a lower bound for $\mathsf{Gap\text{-}MCSP}[n^d, 2^{(\alpha/2-o(1))n}]$ instead of $\mathsf{Gap\text{-}MCSP}[Cn^2, 2^{(\alpha/2)n-2}]$ because of a polynomial circuit complexity overhead in the argument, which is not present in the case of Kt complexity since there one takes the logarithmic of the running time when measuring complexity, and because the circuit complexity (measured by number of gates) of a random string can be slightly smaller than its Kt complexity.

B Hardness Magnification and Proof Complexity

The initial instance of hardness magnification from [41] says that if an average-case version of MCSP (with inputs being truth tables of Boolean functions) is worst-case hard for formulas of super-linear size, then its succinct version (with inputs being lists of input-output tuples representing partial Boolean functions) is hard for NC^1 (cf. [41, Theorem 1]).

Hardness magnification for MCSP thus attacks strong circuit lower bounds by 1. employing the natural proofs barrier which states a conditional hardness of MCSP, and 2. exploiting the difference between feasible (succinct) and infeasible (uncompressed) formulations of a meta-computational problem like MCSP.

This strategy has a history in proof complexity. The work of Razborov [45, 46] and Krajíček [31] formulated the natural proofs barrier as a conditional proof complexity lower bound expressing hardness of tautologies encoding circuit lower bounds. This idea was further developed in the theory of proof complexity generators [30, 2]. It has led, in particular, to Razborov's conjecture [47] about hardness of Nisan-Wigderson generators for strong proof systems. Razborov's conjecture is designed to imply hardness of circuit lower bounds formalized in a way so that the whole truth table of the hard function is hardwired into the formula.

The realization that a feasible formulation of circuit lower bounds should be much harder than the infeasible truth table formulas inspired the result about unprovability of circuit lower bounds in theories of bounded arithmetic such as VNC^1, cf. [43], and the proposal [42, 0.1 Circuit lower bounds and Complexity-Theoretic tautologies] to study exponentially harder lb formulas. Once the definitions are given, it is for example clear that polynomial-size proofs of the lb formulas transform into almost linear-size proofs of the truth table formulas. Another instance of this phenomena says that:

If the truth table formulas encoding a polynomial circuit lower bound require superlinear-size proofs in AC^0-Frege systems, then lb formulas encoding the same polynomial circuit lower bound require (NC^1)-Frege proofs of super-polynomial size (implicit in the proof of [37, Proposition 4.14]).

Since AC^0-Frege lower bounds are known, this suggests a way for attacking Frege lower bounds. ([41] established analogous results in circuit complexity, where it might be easier to prove lower bounds. However, their version of the MCSP problem refers to the average-case complexity of truth-tables, which seems harder to analyse. We refer to [41] for further discussion.)

The lb formulas result from the feasible witnessing of circuit lower bounds. In [37], the witnessing was provided by a theorem of Lipton and Young [34] establishing the existence of anti-checkers, described in Section 1.2. This allows to express the hardness of f without using its whole truth table. The present paper extends the idea of anti-checkers into the context of hardness magnification in circuit complexity for the standard worst-case formulation of MCSP.

Non-Malleable Extractors and Non-Malleable Codes: Partially Optimal Constructions

Xin Li
Department of Computer Science, Johns Hopkins University, Baltimore, MD, USA
http://www.cs.jhu.edu/~lixints/
lixints@cs.jhu.edu

Abstract

The recent line of study on randomness extractors has been a great success, resulting in exciting new techniques, new connections, and breakthroughs to long standing open problems in several seemingly different topics. These include seeded non-malleable extractors, privacy amplification protocols with an active adversary, independent source extractors (and explicit Ramsey graphs), and non-malleable codes in the split state model. Previously, the best constructions are given in [54]: seeded non-malleable extractors with seed length and entropy requirement $O(\log n + \log(1/\epsilon)\log\log(1/\epsilon))$ for error ϵ; two-round privacy amplification protocols with optimal entropy loss for security parameter up to $\Omega(k/\log k)$, where k is the entropy of the shared weak source; two-source extractors for entropy $O(\log n \log\log n)$; and non-malleable codes in the 2-split state model with rate $\Omega(1/\log n)$. However, in all cases there is still a gap to optimum and the motivation to close this gap remains strong.

In this paper, we introduce a set of new techniques to further push the frontier in the above questions. Our techniques lead to improvements in all of the above questions, and in several cases partially optimal constructions. This is in contrast to all previous work, which only obtain close to optimal constructions. Specifically, we obtain:

1. A seeded non-malleable extractor with seed length $O(\log n) + \log^{1+o(1)}(1/\epsilon)$ and entropy requirement $O(\log\log n + \log(1/\epsilon))$, where the entropy requirement is asymptotically optimal by a recent result of Gur and Shinkar [40];
2. A two-round privacy amplification protocol with optimal entropy loss for security parameter up to $\Omega(k)$, which solves the privacy amplification problem completely;[1]
3. A two-source extractor for entropy $O(\frac{\log n \log\log n}{\log\log\log n})$, which also gives an explicit Ramsey graph on N vertices with no clique or independent set of size $(\log N)^{O(\frac{\log\log\log N}{\log\log\log\log N})}$; and
4. The first explicit non-malleable code in the 2-split state model with *constant* rate, which has been a major goal in the study of non-malleable codes for quite some time. One small caveat is that the error of this code is only (an arbitrarily small) constant, but we can also achieve negligible error with rate $\Omega(\log\log\log n/\log\log n)$, which already improves the rate in [54] exponentially.

We believe our new techniques can help to eventually obtain completely optimal constructions in the above questions, and may have applications in other settings.

2012 ACM Subject Classification Theory of computation → Expander graphs and randomness extractors

Keywords and phrases extractor, non-malleable, privacy, codes

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.28

Funding *Xin Li*: Supported by NSF award CCF-1617713.

[1] Except for the communication complexity, which is of secondary concern to this problem.

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka; Article No. 28; pp. 28:1–28:49

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

1 Introduction

The study of randomness extractors has been a central line of research in the area of pseudorandomness, where the goal is to understand how to use randomness more efficiently in computation. As fundamental objects in this area, randomness extractors are functions that transform imperfect random sources into nearly uniform random bits. Their original motivation is to bridge the gap between the uniform random bits required in standard applications (such as in randomized algorithms, distributed computing, and cryptography), and practical random sources which are almost always biased (either because of natural noise or adversarial information leakage). However the study of these objects has led to applications far beyond this motivation, in several different fields of computer science and combinatorics (e.g., coding theory, graph theory, and complexity theory).

The inputs to a randomness extractor are usually imperfect randomness, modeled by the notion of general weak random sources with a certain amount of entropy.

▶ **Definition 1.** *The* min-entropy *of a random variable X is*

$$H_\infty(X) = \min_{x \in \mathsf{supp}(X)} \log_2(1/\Pr[X = x]).$$

For $X \in \{0,1\}^n$, we call X an $(n, H_\infty(X))$-source, and we say X has entropy rate $H_\infty(X)/n$.

An extensively studied model of randomness extractors is the so called *seeded extractors*, introduced by Nisan and Zuckerman [60]. The inputs to a seeded extractor are a general weak random source and a short independent uniform random seed. The random seed is necessary here since it is well known that no deterministic extractor with one general weak source as input can exist. Such extractors have wide applications in computer science.

▶ **Definition 2** (Seeded Extractor)**.** *A function* $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *is a* (k, ε)-extractor *if for every source X with min-entropy k and independent Y which is uniform on $\{0,1\}^d$,*

$$|\mathsf{Ext}(X, Y) - U_m| \leq \epsilon.$$

If in addition we have $|(\mathsf{Ext}(X, Y), Y) - (U_m, Y)| \leq \epsilon$ then we say it is a strong (k, ε)-extractor.

Through a long line of research, we now have explicit constructions of seeded extractors with almost optimal parameters (e.g., [55, 41, 33, 32]). In the last decade or so, the focus has shifted to several different but related models of randomness extractors, including seedless extractors and non-malleable extractors. The study of these topics has also been quite fruitful, leading to breakthroughs to several long standing open problems.

1.1 Seedless extractors

As the name suggests, a seedless extractor uses no uniform seed, and the only inputs are weak random sources. Here, again we have two different cases. In the first case, one puts additional restrictions on a single weak random source in order to allow possible extraction, thus obtaining deterministic extractors for special classes of (structured) sources. In the second case, the sources are still general weak random sources, but the extractor needs to use more than one sources. To make extraction possible, one typically assumes the input sources to the extractor are independent, and this kind of extractors are sometimes called independent source extractors.

Since the pioneering work of Chor and Goldreich [19], the study of independent source extractors has gained significant attention due to their close connections to explicit Ramsey graphs, and their applications in distributed computing and cryptography with general weak random sources [43, 42]. The goal here is to give explicit constructions that match the probabilistic bound: an extractor for just two independent (n, k) sources with $k \geq \log n + O(1)$ that outputs $\Omega(k)$ bits with exponentially small (in k) error. Note that an explicit two-source extractor for such entropy (even with one bit output and constant error) will give an (strongly) explicit Ramsey graph on N vertices with no clique or independent set of size $O(\log N)$, solving an open problem proposed by Erdős [37] in his seminal paper that inaugurated the probabilistic method.

While early progress on this problem has been quite slow, with the best known construction in almost 20 years only able to handle two independent (n, k) sources with $k > n/2$ [19], since 2004 there has been a long line of work [4, 5, 62, 10, 61, 6, 46, 48, 50, 49, 52, 20, 16, 53, 26, 13, 21, 8, 24, 54] introducing exciting new techniques to this problem. This line of work greatly improved the situation and led to a series of breakthroughs. Now we have three source extractors for entropy $k \geq \mathsf{polylog}(n)$ that output $\Omega(k)$ bits with exponentially small error [52, 7], two-source extractors for entropy $k \geq \mathsf{polylog}(n)$ that output $\Omega(k)$ bits with polynomially small error [16, 53, 57], and two-source extractors for entropy $k \geq O(\log n \log\log n)$ that output one bit with any constant error [54]. This also gives an explicit Ramsey graph on N vertices with no clique or independent set of size $(\log N)^{O(\log\log\log N)}$. Interestingly and somewhat surprisingly, the most recent progress which brought the entropy requirement close to optimal, has mainly benefited from the study of another kind of extractors, the so called *non-malleable extractors*, which we now describe below.

1.2 Non-malleable extractors

Non-malleable extractors are strengthening of standard extractors, where one requires that the output is close to uniform even given the output of the extractor on tampered inputs.

▶ **Definition 3** (Tampering Funtion). *For any function $f : S \to S$, We say f has no fixed points if $f(s) \neq s$ for all $s \in S$. For any $n > 0$, let $\mathcal{F}_n$ denote the set of all functions $f : \{0,1\}^n \to \{0,1\}^n$. Any subset of $\mathcal{F}_n$ is a family of tampering functions.*

Depending on what the tampering function acts on, we also have different models of non-malleable extractors. If the tampering acts on the seed of a seeded extractor, such extractors are called *seeded non-malleable extractors*, originally introduced by Dodis and Wichs [30].

▶ **Definition 4.** *A function* $\mathrm{snmExt} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *is a seeded non-malleable extractor for min-entropy k and error ϵ if the following holds: If X is an (n, k) source and $\mathcal{A} : \{0,1\}^d \to \{0,1\}^d$ is an arbitrary tampering function with no fixed points, then*

$$|\mathrm{snmExt}(X, U_d) \circ \mathrm{snmExt}(X, \mathcal{A}(U_d)) \circ U_d - U_m \circ \mathrm{snmExt}(X, \mathcal{A}(U_d)) \circ U_d| < \epsilon$$

where U_m is independent of U_d and X.

If the tampering acts on the sources of an independent source extractor, then we have *seedless non-malleable extractors*, originally introduced by Cheraghchi and Guruswami [18].

▶ **Definition 5.** *A function* $\mathsf{nmExt} : (\{0,1\}^n)^C \to \{0,1\}^m$ *is a* (k,ϵ)*-seedless non-malleable extractor for* C *independent sources, if it satisfies the following property: Let* $X_1, \cdots, X_C$ *be* C *independent* (n,k) *sources, and* $f_1, \cdots, f_C : \{0,1\}^n \to \{0,1\}^n$ *be* C *arbitrary tampering functions such that there exists an* f_i *with no fixed points,*[2] *then*

$$|\mathsf{nmExt}(X_1, \cdots, X_C) \circ \mathsf{nmExt}(f_1(X_1), \cdots, f_C(X_2)) - U_m \circ \mathsf{nmExt}(f_1(X_1), \cdots, f_C(X_2))| < \epsilon.$$

Seeded non-malleable extractors and privacy amplification

Seeded non-malleable extractors were introduced by Dodis and Wichs [30], to study the basic problem of *privacy amplification* [9]. Consider the situation where two parties with local (non-shared) uniform random bits try to convert a shared secret weak random source $\mathbf{X}$ into shared secret uniform random bits. They do this by communicating through a channel, which is watched by an adversary with unlimited computational power. Standard strong seeded extractors provide very efficient protocols for a passive adversary (i.e., can only see the messages but cannot change them), but fail for an active adversary (i.e., can arbitrarily change, delete and reorder messages). In the latter case, which is the focus of this paper, the main goal is to design a protocol that uses as few number of interactions as possible, and achieves a shared uniform random string $\mathbf{R}$ which has *entropy loss* (the difference between the length of the output and $H_\infty(\mathbf{X})$) as small as possible. Such a protocol is defined with a security parameter s, which means the probability that an active adversary can successfully make the two parties output two different strings without being detected is at most 2^{-s}. On the other hand, if the adversary remains passive, then the two parties should achieve a shared secret string that is 2^{-s}-close to uniform. We refer the reader to [29] for a formal definition.

A long line of work has been devoted to this problem [56, 27, 30, 63, 44, 11, 29, 25, 47, 48, 51, 12, 22, 23, 13, 21, 24, 54]. It is known that one round protocol can only exist when the entropy rate of $\mathbf{X}$ is bigger than $1/2$, and the protocol has to incur a large entropy loss. When the entropy rate of $\mathbf{X}$ is smaller than $1/2$, [30] showed that any protocol has to take at least two rounds with entropy loss at least $\Omega(s)$. Achieving a two-round protocol with entropy loss $O(s)$ for all possible security parameters s is thus the holy grail of this problem (note that s can be at most $\Omega(k)$ where $k = H_\infty(\mathbf{X})$).

While early works on this problem used various techniques, in [30], Dodis and Wichs introduced a major tool, the seeded non-malleable extractor defined above. They showed that two-round privacy amplification protocols with optimal entropy loss can be constructed using explicit seeded non-malleable extractors. Furthermore, non-malleable extractors exist when $k > 2m + 2\log(1/\varepsilon) + \log d + 6$ and $d > \log(n-k+1) + 2\log(1/\varepsilon) + 5$. Since then, the study of non-malleable extractors has seen significant progress starting from the first explicit construction in [29], with further connections to independent source extractors established in [48, 50, 16]. Previous to this work, the best known seeded non-malleable extractor is due to the author [54], which works for entropy $k \geq O(\log n + \log(1/\epsilon)\log\log(1/\epsilon))$ and has seed length $d = O(\log n + \log(1/\epsilon)\log\log(1/\epsilon))$. Although close to optimal, the extra $O(\log\log(1/\epsilon))$ factor in the entropy requirement implies that by using this extractor, one can only get two-round privacy amplification protocols with optimal entropy loss for security parameter up to $s = \Omega(k/\log k)$. This still falls short of achieving the holy grail, and may be problematic for some applications. For example, even if the shared weak source has slightly super-logarithmic entropy, the error of the protocol can still be sub-polynomially large; while

[2] The original definition of seedless non-malleable independent source extractors in [18] allows fixed points, but the two definitions are equivalent up to a small loss in parameters. See Section 7 for details.

ideally one can hope to get negligible error, which is important for other cryptographic applications based on this. The only previous protocol that can achieve security parameter up to $s = \Omega(k)$ is the work of [11], which has entropy loss $O(\log n + s)$ but also uses $O(\log n + s)$ rounds of interactions, much larger than 2. This also results in a total communication complexity of $O((\log n + s)^2)$ and requires the two parties' local random bits to be at least this long.

Seedless non-malleable extractors and non-malleable codes

Seedless non-malleable extractors were first introduced by Cheraghchi and Guruswami [18] to study non-malleable codes [36], a generalization of standard error correcting codes to handle a much larger class of attacks. Informally, a non-malleable code is defined w.r.t. a specific family of tampering functions $\mathcal{F}$. The code consists of a randomized encoding function E and a deterministic decoding function D, such that for any $f \in \mathcal{F}$, if a codeword $E(x)$ is modified into $f(E(x))$, then the decoded message $x' = D(f(E(x)))$ is either the original message x or a completely unrelated message. The formal definition is given in Section 7. In [36], Dziembowski et. al showed that such codes can be used generally in tamper-resilient cryptography to protect the memory of a device.

Even with such generalization, non-malleable codes still cannot exist if $\mathcal{F}$ is completely unrestricted. However, they do exist for many broad families of tampering functions. One of the most studied families of tampering functions is the so called *t-split-state* model. Here, a k-bit message x is encoded into a codeword with t parts $y_1, \cdots, y_t$, each of length n. An adversary can then arbitrarily tamper with each y_i independently. In this case, the rate of the code is defined as $k/(tn)$.

This model arises naturally in many applications, typically when different parts of memory are used to store different parts of $y_1, \cdots, y_t$. Such a code can also be viewed as a kind of "non-malleable secret sharing scheme". The case of $t = 2$ is the most useful and interesting setting, since $t = 1$ corresponds to the case where $\mathcal{F}$ is unrestricted. Again, there has been a lot of previous work on non-malleable codes in this model. In this paper we will focus on the information theoretic setting.

Dziembowski et. al [36] first proved the existence of non-malleable codes in the split-state model. Cheraghchi and Guruswami [17] showed that the optimal rate of such codes in the 2-split-state model is 1/2. Since then a major goal is to construct explicit non-malleable codes in the 2-split-state model with constant rate. The first construction appears in [34], with later improvements in [3, 2, 1], but all constructions only achieve rate $n^{-\Omega(1)}$.

Cheraghchi and Guruswami [18] found a way to construct non-malleable codes in the t-split state model using non-malleable t-source extractors. Chattopadhyay and Zuckerman [15] constructed the first seedless non-malleable extractor, which works for 10 independent sources with entropy $(1-\gamma)n$, and consequently they obtained a constant rate non-malleable code in the 10-split-state model. Subsequently, constructions of non-malleable two source extractors appeared in [12] and [54]. Both constructions work for min-entropy $k = (1-\gamma)n$, and the former gives a non-malleable code in the 2-split state model with rate $n^{-\Omega(1)}$ while the latter achieves rate $\Omega(\frac{1}{\log n})$. Very recently, a work by Kanukurthi et. al [45] achieved constant rate in the 4-split state model, and another one by Gupta et. al [39] achieved constant rate in the 3-split state model, but the best construction in the 2-split state model still only achieves rate $\Omega(\frac{1}{\log n})$ [54].

As can be seen from the above discussions, extensive past research has established strong connections among these different topics, and provided solutions close to optimal. However, there still remains a gap and the motivation to close this gap remains strong.

We also remark about the curious coincidences of some parameters here and the parameters in the literature of constructing unconditional pseudorandom generators for small space computation, another central line of research in the area of pseudorandomness. The holy grail in this case is to construct a logspace explicit pseudorandom generator (a function that stretches a short uniform random seed into a long string that looks random) which fools logspace computation with seed length $O(\log(n/\epsilon))$ and error ϵ. Such a construction would imply that randomness does not add any more power in logspace computation. Although for general logspace computation the best known pseudorandom generator remains Nisan's generator [59] which needs seed length $O(\log n \log(n/\epsilon))$, recently there have been several works achieving near optimal seed length for restricted cases. For example, Meka et al. [58] constructed a pseudorandom generator for width-3 branching programs with seed length $O(\log(n/\epsilon)\mathsf{poly}\log\log(n/\epsilon)) + O(\log n \log(1/\epsilon))$, and Doron et al. [31] constructed a pseudorandom generator for constant depth read once formulas with seed length $O(\log(n/\epsilon)\mathsf{poly}\log\log(n/\epsilon))$, improving a previous similar result for depth 2 read-once formulas by Gopalan et al. [38]. The extra $\mathsf{poly}\log\log(1/\epsilon)$ dependence on error ϵ is quite similar to the previously best known non-malleable extractors, and in the case of constant or polynomially small error the seed length becomes $\log n\mathsf{poly}\log\log n$, which is again quite similar to the entropy requirement of the previously best known two-source extractors.

The high level reason for these coincidences is that all constructions use some recursive steps (e.g., for $O(\log\log n)$ steps), where in each step one needs to use say $O(\log n)$ *independent* random bits, and thus the total entropy requirement becomes at least $\Omega(\log n \log\log n)$. Circumventing this barrier needs new techniques and can lead to improved or potentially optimal constructions, which is of great interests. In this sense, the results in this paper are the first kind to break this barrier, and we believe that the set of new techniques we introduce can lead to further improvements and potentially optimal constructions.

1.3 Our Results

In this paper we achieve improvements in all the questions discussed in the context of extractors, and in several cases partially optimal constructions. In contrast, all previous works only obtain close to optimal constructions. Our first theorem gives explicit seeded non-malleable extractors with optimal entropy requirement.

▶ **Theorem 6.** *There exists a constant $C > 1$ such that for any constant $a \in \mathcal{N}, \dashv \geq \in$, any $n, k \in \mathcal{N}$ and any $0 < \epsilon < 1$ with $k \geq C(\log\log n + a\log(1/\epsilon))$, there is an explicit construction of a strong seeded (k, ϵ) non-malleable extractor $\{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $d = O(\log n) + \log(1/\epsilon)2^{O(a(\log\log(1/\epsilon))^{\frac{1}{a}})}$ and $m = \Omega(k)$.*

Note that this theorem provides a trade-off between the entropy requirement and the seed length. For example, if we take $a = 2$, then the entropy requirement is $O(\log\log n + \log(1/\epsilon))$ while the seed length is $O(\log n) + 2^{O(\sqrt{\log\log(1/\epsilon)})}\log(1/\epsilon) = O(\log n) + \log^{1+o(1)}(1/\epsilon)$. By a recent result of Gur and Shinkar [40], the entropy requirement in our construction is asymptotically optimal. We can also achieve smaller seed length while requiring slightly larger entropy.

▶ **Theorem 7.** *There exists a constant $C > 1$ such that for any $n, k \in \mathcal{N}$ and $0 < \epsilon < 1$ with $k \geq C(\log\log n + \log(1/\epsilon)\log\log\log(1/\epsilon))$, there is an explicit construction of a strong seeded (k, ϵ) non-malleable extractor $\{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $d = O(\log n + \log(1/\epsilon)(\log\log(1/\epsilon))^2)$[3] and $m = \Omega(k)$.*

[3] The exponent 2 can be reduced to be arbitrarily close to $\log 3$.

Combined with the protocol in [30], we have the following theorem.

▶ **Theorem 8.** *There exists a constant $0 < \alpha < 1$ such that for any $n, k \in \mathcal{N}$, there is an explicit two-round privacy amplification protocol in the presence of an active adversary that achieves (1) any security parameter $s \leq \alpha k$, entropy loss $O(\log\log n + s)$ and communication complexity $O(\log n) + s2^{O(a(\log s)^{\frac{1}{a}})}$ for any constant integer $a \geq 2$, or (2) any security parameter $s \leq \alpha k/\log\log k$, entropy loss $O(\log\log n + s)$ and communication complexity $O(\log n + s\log^2 s)$.*

Our two-round protocol can achieve optimal entropy loss for security parameter up to $s = \Omega(k)$, thus achieving the holy grail of this problem. Compared to the $O(\log n + s)$-round protocol in [11], our protocol also has better dependence on n and significantly better communication complexity.

We remark that the $O(\log\log n)$ term is also the best possible (up to constant) if one wants to apply the two-round protocol in [30]. This is because the output of the non-malleable extractor is used as the key for a message authentication code (MAC) that authenticates the seed of a strong seeded extractor with security parameter s. Since the seed of the extractor uses at least $\Omega(\log n)$ bits, the MAC requires a key of length at least $\log\log n + s$. See [30] for more details.

▶ Remark 9. In Theorem 6 and Theorem 7, the dependence on error ϵ in the seed length and the entropy requirement can be switched. For example, in Theorem 6, we can also achieve $k \geq C\log\log n + \log(1/\epsilon)2^{C\cdot a(\log\log(1/\epsilon))^{\frac{1}{a}}}$ and $d = O(\log n + a\log(1/\epsilon))$, i.e., we can achieve asymptotically optimal parameters in either the seed length or the entropy requirement, but not in both.

We also have the following non-malleable two-source extractor and seeded non-malleable extractor.

▶ **Theorem 10.** *There exists a constant $0 < \gamma < 1$ and a non-malleable two-source extractor for $(n, (1-\gamma)n)$ sources with error $2^{-\Omega(n\log\log n/\log n)}$ and output length $\Omega(n)$.*

▶ **Theorem 11.** *There is a constant $C > 0$ such that for any $\epsilon > 0$ and $n, k \in \mathcal{N}$ with $k \geq C(\log\log n + \frac{\log(1/\epsilon)\log\log(1/\epsilon)}{\log\log\log(1/\epsilon)})$, there is an explicit strong seeded non-malleable extractor for (n, k) sources with seed length $d = O(\log n + \frac{\log(1/\epsilon)\log\log(1/\epsilon)}{\log\log\log(1/\epsilon)})$, error ϵ and output length $\Omega(k)$.*

Combined with the techniques in [8], we obtain the following theorems.

▶ **Theorem 12.** *For every constant $\epsilon > 0$, there exists a constant $C > 1$ and an explicit two source extractor $\mathsf{Ext} : (\{0,1\}^n)^2 \to \{0,1\}$ for entropy $k \geq C\frac{\log n\log\log n}{\log\log\log n}$ with error ϵ.*

▶ **Corollary 13.** *For every large enough integer N there exists a (strongly) explicit construction of a K-Ramsey graph on N vertices with $K = (\log N)^{O(\frac{\log\log\log N}{\log\log\log\log N})}$.*

Our result gives the first two-source extractor for entropy $o(\log n\log\log n)$ and the first explicit K-Ramsey graph on N vertices with $K = (\log N)^{o(\log\log\log N)}$. For non-malleable codes in the 2-split state model, we have the following theorem.

▶ **Theorem 14.** *There are constants $0 < \eta, \mu < 1$ such that for any $n \in \mathcal{N}$ and $2^{-\frac{\mu n}{\log n}} \leq \epsilon \leq \eta$ there exists an explicit non-malleable code in the 2-split-state model with block length $2n$, rate $\Omega(\frac{\log\log\log(1/\epsilon)}{\log\log(1/\epsilon)})$ and error ϵ.*

Note that if we choose $\epsilon = 2^{-c}$ for some constant $c > 1$, then we get a non-malleable code with rate $\Omega(\frac{\log\log c}{\log c})$ and error 2^{-c}. This gives the first construction of an explicit non-malleable code in the 2-split-state model with *constant* rate. Note that the error can be arbitrarily small, and the dependence of the rate on the error is pretty good. For example, even if one wants to achieve error $2^{-2^{100}}$, which is more than enough for any practical application, the rate is on the order of 1/16. On the other hand, if we choose $\epsilon = 2^{-\mathsf{polylog}(n)}$, then we get a non-malleable code with negligible error and rate $\Omega(\frac{\log\log\log n}{\log\log n})$, which already improves the rate in [54] exponentially.

We can also achieve close to exponentially small error with an improved rate.

▶ **Theorem 15.** *For any $n \in \mathcal{N}$ there exists a non-malleable code with efficient encoder-/decoder in the 2-split-state model with block length $2n$, rate $\Omega(\log\log n/\log n)$ and error $\epsilon = 2^{-\Omega(n\log\log n/\log n)}$.*

1.4 Overview of The Constructions and Techniques

We demonstrate our techniques here by an informal overview of our constructions. Throughout this section we will be mainly interested in the dependence of various parameters (e.g., seed length, entropy requirement) on the error ϵ, since this makes the presentation cleaner. The dependence on n comes from the alternating extraction between the seed and the source, thus the seed needs to have an $O(\log n)$ term while the source only needs an $O(\log\log n)$ term.

All recent constructions of non-malleable extractors essentially follow the same high level sketch: first obtain a small advice on $L = O(\log(1/\epsilon))$ bits such that with probability $1-\epsilon$, the advice is different from its tampered version. Then, a correlation breaker with advice (informally introduced in [12] and formally defined in [22]) is used to obtain the final output. A correlation breaker with advice is a function $\mathsf{AdvCB} : X \times Y \times \alpha \to V$ where X, Y are two independent sources (in the case of a seeded non-malleable extractor, Y can be viewed as the seed), such that if the advice α is not equal to its tampered version α', then the output $\mathsf{AdvCB}(X, Y, \alpha)$ is close to uniform conditioned on the tampered version $\mathsf{AdvCB}(X', Y', \alpha')$. There are several constructions of correlation breakers, and the goal is to minimize the entropy requirement of X and Y. Previously, the most efficient construction is based on a non-malleable independence preserving merger (NIPM for short, introduced in [26] and generalized in [13]) in [54], which achieves entropy requirement $O(\log L\log(1/\epsilon))$ using a recursive structure. As discussed before, improving this needs new techniques, and our main new idea is the following:

Idea 1: *Instead of using fresh randomness, we borrow techniques from pseudorandom generators for small space computation [59, 60] to* recycle *the randomness in each step of the independence preserving merger. In this sense, we construct* pseudorandom *independence preserving mergers.*

Using this idea, we can improve the entropy requirement of X and Y in two different settings. In the asymmetric case, one of them can be optimal while the other is larger, e.g., one can be $O(\log(1/\epsilon))$ and the other is $2^{O(\sqrt{\log L})}\log(1/\epsilon)$. This is good for seeded non-malleable extractors and privacy amplification protocols. In the symmetric case which is needed for two-source extractors, one can reduce the entropy requirement of both X and Y to $O(\frac{\log L}{\log\log L}\log(1/\epsilon))$.

Turning to achieving constant rate non-malleable codes, our main new idea is the following:

Idea 2: *In the connection between non-malleable codes and non-malleable two-source extractors found by Cheraghchi and Guruswami [18], the way of dealing with errors is* too coarse. *For most recent constructions of non-malleable two-source extractors, one can actually separate two different kinds of error, where one kind is relevant and the other is* irrelevant.

We now discuss both ideas in more details. We start with Idea 1 and first briefly recall the construction of the merger in [54]. The NIPM takes an $L \times m$ random matrix V with $m = O(\log(1/\epsilon))$ with the property that one row in the matrix is uniform given the corresponding row in its tampered version[4] (this can be obtained from the advice and inputs), and does the following. Suppose the matrix V is a deterministic function of the source X, then we first generate $\ell = \log L$ random variables $(Y_1, \cdots, Y_\ell)$ from Y, such that each Y_i is close to uniform given the previous random variables and their tampered versions (i.e., $(Y_1, Y_1', \cdots, Y_{i-1}, Y_{i-1}')$). We call this property the *look-ahead* property. Next, we run a simple merger for ℓ iterations, with each iteration using a new Y_i to merge every two consecutive rows in V, thus decreasing the number of rows by a factor of 2. We output the final matrix V which has one row.

Let's turn to the entropy requirement. In this construction each Y_i needs to have at least $\Omega(\log(1/\epsilon))$ bits in order to ensure the error is at most ϵ, thus it is clear that Y needs to have entropy at least $\Omega(\ell \log(1/\epsilon)) = \Omega(\log(1/\epsilon) \log\log(1/\epsilon))$. However, it turns out that X also needs to have such entropy, for the following two reasons. First, in each iteration after we apply the simple merger, the length of each row in the matrix decreases by a constant factor (due to the entropy loss of any seeded extractor). Thus we cannot afford to just repeat the process for ℓ times since that would require the original row in V (and hence X) to have entropy at least $\mathsf{polylog}(1/\epsilon)$. Instead, we again create ℓ random variables $(X_1, \cdots, X_\ell)$ from X with the look-ahead property, and in each iteration after merging we use each row of the matrix to extract from a new X_i (using a standard seeded extractor, and possibly after first extracting from another new Y_i), to restore the length of the rows in the matrix. We need the look-ahead property in $(X_1, \cdots, X_\ell)$ and $(Y_1, \cdots, Y_\ell)$ so that after each iteration we can fix the previously used random variables and maintain the independence of X and Y, as well as the fact that the matrix is a deterministic function of X. Each X_i again needs at least $\Omega(\log(1/\epsilon))$ bits so this puts a lower bound on the entropy of X.

Second, in order to prepare the random variables $(Y_1, \cdots, Y_\ell)$, we in fact run an alternating extraction protocol between (part of) X and Y. This protocol lasts 2ℓ rounds between X and Y, and in each round either X or Y needs to spend $\Omega(\log(1/\epsilon))$ random bits. This again puts a lower bound of $\Omega(\ell \log(1/\epsilon))$ on the entropy of X.

We remark that the above description is slightly different from the standard definition of an NIPM, where the only input besides the matrix V is Y. Indeed, in [54] it was presented as a correlation breaker. However, these two objects are actually similar, and for this paper it is more convenient to consider NIPMs with an additional input X, which is independent of Y but may be correlated with V. We will use this notion here and formally define it in Section 4.

Improved merger construction

To break the above barriers, our key observation is that we can *recycle* the entropy in X, similar in sprit to what has been done in previous constructions of pseudorandom generators for small space computation [59, 60]. Indeed, the random variables $(X_1, \cdots, X_\ell)$ can be

[4] Sometimes we also require the other rows to be uniform, in order to make the construction simpler. This is the case of this paper, but we ignore the issue here for simplicity and clarity.

replaced by the original source X, as long as we have slightly more (e.g., 2ℓ) Y_i's and they satisfy the look ahead property. To achieve this we crucially use the property that the NIPM only needs one row of V to be uniform given the corresponding row in its tampered version, and does not care about the dependence among the rows of V (they can have arbitrary dependence). Consider a particular iteration i in which we have just finished applying the simple merger. We can first fix all random variables $\{Y_j\}$ that have been used so far, and conditioned on this fixing we know that X and Y are still independent, and the matrix V is a deterministic function of X, which is independent of all random variables obtained from Y. To restore the length of each row in V, we use each row of V to first extract $O(\log(1/\epsilon))$ bits from Y_{j+1}, and then extract back from the original source X. Note that we only need to consider each row separately (since we don't care about the dependence among them). Assume row h in V has the property that V_h is uniform given V'_h (the tampered version). Since each random variable only has $O(\log(1/\epsilon))$ bits, as long as the entropy of X is $c\log(1/\epsilon)$ for a large enough constant $c > 1$, we can argue that conditioned on the fixing of (V_h, V'_h), X still has entropy at least some $O(\log(1/\epsilon))$. On the other hand since V_h is uniform given V'_h, their corresponding outputs after extracting from (Y_{j+1}, Y'_{j+1}) will also preserve this independence; and conditioned on the fixing of (V_h, V'_h), these outputs are deterministic functions of (Y, Y'), which are independent of (X, X'). Thus they can be used to extract back from (X, X') and preserve the independence. By standard properties of a strong seeded extractor, this holds even conditioned on the fixing of (Y_{j+1}, Y'_{j+1}). Note that conditioned on the further fixing of (Y_{j+1}, Y'_{j+1}), the new matrix is again a deterministic function of X, thus we can go into the next iteration. Therefore, by recycling the entropy in X, altogether we only need X to have entropy some $O(\log(1/\epsilon))$. In each iteration we use two new Y_i's so we need roughly 2ℓ such random variables.

However, we still need to address the second problem, where we need to generate the random variables $(Y_1, \cdots, Y_{2\ell})$. The old way to generate them by using an alternating extraction protocol requires entropy roughly $O(\ell \log(1/\epsilon))$ from X. To solve this problem, we develop a new approach that requires much less entropy from X. For simplicity assume that Y is uniform, we first take 2ℓ slices Y^i from Y, where Y^i has size $(2^i - 1)d$ for some $d = O(\log(1/\epsilon))$. This ensures that even conditioned on the fixing of $(Y^1, Y'^1, \cdots, Y^{i-1}, Y'^{i-1})$, the (average) conditional min-entropy of Y_i is at least $(2^i - 1)d - 2 \cdot (2^{i-1} - 1)d = d$. Then, we can take $O(\log(1/\epsilon))$ uniform bits obtained from X, and use the *same* bits to extract Y_i from Y^i for every i. As long as we use a strong seeded extractor here, we are guaranteed that $(Y_1, \cdots, Y_{2\ell})$ satisfy the look-ahead property; and moreover conditioned on the fixing of the $O(\log(1/\epsilon))$ bits from X, we have that $(Y_1, \cdots, Y_{2\ell})$ is a deterministic function of Y. Note here again we only require entropy $O(\log(1/\epsilon))$ from X, and together with the approach described above this gives us a non-malleable extractor where X can have entropy $O(\log(1/\epsilon))$. However Y will need to have entropy at least $2^{2\ell}O(\log(1/\epsilon)) = O(\log^3(1/\epsilon))$.

To improve the entropy requirement of Y, we note that in the above approach, we only used part of X once to help obtaining the $\{Y^i\}$. Thus we have to use larger and larger slices of Y which actually waste some entropy. Instead, we can use several parts of X, each with $O(\log(1/\epsilon))$ uniform bits. For example, suppose that we have obtained X^1 and X^2, where each is uniform on some $O(\log(1/\epsilon))$ bits and X^2 is uniform even conditioned on the fixing of (X^1, X'^1). We can now take some t slices $\{Y^i\}$ of Y, each of length $(2^i - 1) \cdot 2d$ for some parameters t, d. We first use X^1 to extract from each Y^i and obtain d uniform bits. Note that conditioned on the fixing of (X^1, X'^1), these t random variables already satisfy the look-ahead property. Now for each of these d bits obtained from Y^i, we can apply the same process, i.e., we take some t slices of these d bits, each of length $(2^i - 1) \cdot O(\log(1/\epsilon))$ and then use

X^2 to extract from each of them. This way we obtain t^2 random variables $\{Y_i\}$ that satisfy the look-ahead property. We can thus choose $t^2 = 2\ell$ which means $t = O(\sqrt{\ell})$. The entropy requirement of Y is roughly $(2^t - 1) \cdot (2^t - 1)O(\log(1/\epsilon)) = O(2^{2t} \log(1/\epsilon)) = 2^{O(\sqrt{\ell})} \log(1/\epsilon)$, while the entropy requirement for X is $2 \cdot O(\log(1/\epsilon)) + O(\log(1/\epsilon)) = O(\log(1/\epsilon))$. This significantly improves the entropy requirement of Y.

We can repeat the previous process and use some a parts $(X^1, \cdots, X^a)$ obtained from X. As long as a is a constant, X only needs entropy $O(a \log(1/\epsilon)) = O(\log(1/\epsilon))$, while the entropy requirement of Y is reduced to $2^{O(a\ell^{\frac{1}{a}})} \log(1/\epsilon) = 2^{O(a \log\log(1/\epsilon)^{\frac{1}{a}})} \log(1/\epsilon)$. To prepare the a parts of X, we perform an initial alternating extraction between X and Y, which only needs entropy $O(a \log(1/\epsilon))$ from either of them. This gives Theorem 6. In the extreme case, we can try to minimize the entropy requirement of Y by first creating $\log \ell + 1 = \log\log\log(1/\epsilon) + O(1)$ X^i's, and in each step using a new X^i to double the number of Y_i's. This can be done by using the same X^i to do an alternating extraction of two rounds with each Y_i in parallel. Thus after $\log \ell + 1$ steps we obtain $(Y_1, \cdots, Y_{2\ell})$. Now X needs to have entropy $O(\log(1/\epsilon) \log\log\log(1/\epsilon))$. Ideally, we would want to claim that Y needs entropy $O(\log(1/\epsilon) \log\log(1/\epsilon))$, but due to technical reasons we can only show that this works as long as Y has entropy $O(\log(1/\epsilon)(\log\log(1/\epsilon))^2)$.

The balanced case

In the above discussion, the entropy requirement for X and Y is unbalanced, in the sense that one of them can be quite small, while the other is relatively large. For applications to two-source extractors and non-malleable codes, we need a balanced entropy requirement. Upon first look it does not seem that our new techniques can achieve any improvement in this case, since we are still merging two rows of the matrix V in each step, and for this merging we need at least $\Omega(\log(1/\epsilon))$ fresh random bits. Note that we need $\ell = \log L = \log\log(1/\epsilon)$ steps to finish the merging, thus it seems the total entropy requirement is at least $\Omega(\log(1/\epsilon) \log\log(1/\epsilon))$.

Our key observation here is that we can again apply the idea of recycling entropy. Specifically, let us choose a parameter $t \in \mathcal{N}$ and we merge every t rows in the matrix V at each step, using some merger that we have developed above. For example, we can choose the merger which for merging t rows, requires X to have entropy $O(\log(1/\epsilon))$ and Y to have entropy $2^{O(\sqrt{\log t})} \log(1/\epsilon)$. This will take us $\frac{\log L}{\log t}$ steps to finish merging, and we will do it in the following way. First, we create $s = O(\frac{\log L}{\log t})$ random variables $X_1, \cdots, X_s$ that satisfy the look-ahead property. Then, in each step of the merging, we will use a new X_j. The X_j's can be prepared by taking a small slice of both X and Y and do an alternating extraction protocol with $O(s)$ rounds, which consumes entropy $O(s \log(1/\epsilon)) = O(\frac{\log L}{\log t} \log(1/\epsilon))$ from both X and Y. However, in each step of the merging, we will *not* use fresh entropy from Y, but will recycle the entropy in Y. Note that by doing this, we are recycling the entropy in both X and Y. The recycling in X is done within each step of applying the small merger, while the recycling in Y is done between these steps.

Now, consider a particular step i in the merging. Since we are using a new X_j in each step, we can fix all previous X_j's that have been used and their tampered versions. Conditioned on this fixing, the matrix V obtained so far (and the tampered version V') is a deterministic function of Y, therefore independent of X. We now want to claim that conditioned on the random variable (V, V'), Y still has high entropy. If this is true then we can take a new X_{j+1} and apply a strong seeded extractor to Y using X_{j+1} as the seed, and the extracted random bits (which are deterministic functions of Y conditioned on the fixing of X_{j+1}) can be used for merging in the next step. Also note that to apply the merger, we can take yet another

new X_{j+2} and use each row of V to extract from X_{j+2} and create a matrix W. Conditioned on the fixing of (V, V'), we have that (W, W') is a deterministic function of (X, X') and therefore independent of (Y, Y'). Moreover the independence between corresponding rows in (V, V') is preserved in (W, W') (i.e., there is also a row in W that is uniform given the corresponding row in W'). Thus now we can indeed apply the merger again to W and the extracted random bits from Y, possibly together with a new X_{j+3}. Again, this is similar in spirit to what has been done in previous constructions of pseudorandom generators for small space computation [59, 60].

The above idea indeed works, except for the following subtle point: in the first several steps of merging, the matrix V can have many rows and the size of V can be larger than the entropy of Y, unless Y has entropy $\Omega(\log^2(1/\epsilon))$. Thus conditioning on (V, V') may cause Y to lose all entropy. To get around this, we again use the fact that we only need one row in V to be independent of the corresponding row in V' (call this the good row), and does not care about the dependence between different rows. Thus in each step, we only need to condition on the fixing of the t rows that we are merging (and their tampered versions). This ensures that if originally there is a good row in these t rows, then after merging the output is also a good row in the new matrix. Thus, we only need the entropy of Y to be $O(t\log(1/\epsilon)) + 2^{O(\sqrt{\log t})}\log(1/\epsilon) + O(\frac{\log L}{\log t}\log(1/\epsilon)) = O(t\log(1/\epsilon) + \frac{\log L}{\log t}\log(1/\epsilon))$ since we will maintain the length of each row in V to be $O(\log(1/\epsilon))$. Now by choosing $t = \frac{\log L}{\log\log L}$, both X and Y only need entropy $O(\frac{\log L}{\log\log L}\log(1/\epsilon)) = O(\frac{\log(1/\epsilon)\log\log(1/\epsilon)}{\log\log\log(1/\epsilon)})$. By the connections in [54, 8, 18], this dependence gives Theorem 10, 11, 12 and 15.

Non-malleable codes

To further improve the rate of non-malleable codes in the 2-split state model, we re-examine the connection between non-malleable codes and non-malleable two-source extractors found by Cheraghchi and Guruswami [18]. They showed that given a non-malleable two-source extractor with error ϵ and output length m, the uniform sampling of the pre-image of any given output gives an encoding of a non-malleable code in the 2-split state model with error roughly $2^m\epsilon$. This blow up of error comes from the conditioning on the event that the output of the extractor is a given string in $\{0,1\}^m$, which roughly has probability 2^{-m}. Therefore, one needs $m < \log(1/\epsilon)$, and thus the error of the extractor puts a limit on the rate of the code.

To break this barrier, we note that all recent constructions of non-malleable two-source extractors [12, 54] follow a very special framework. As mentioned before, these constructions first obtain an advice $\tilde{\alpha}$ such that with probability $1-\epsilon_1$ we have $\tilde{\alpha} \neq \tilde{\alpha}'$, where $\tilde{\alpha}'$ is the tampered version. Then, using a correlation breaker with advice one obtains the output. This part has error ϵ_2, and the final error of the extractor is $\epsilon_1 + \epsilon_2$.

In all previous work, this error is treated as a whole, but our key observation here is that these two errors ϵ_1 and ϵ_2 can actually be treated *separately*. More specifically, the error that matters most for the rate of the code is actually ϵ_2, not ϵ_1. Intuitively, this is because the event $\tilde{\alpha} \neq \tilde{\alpha}'$ is determined by a set of random variables that have small size compared to the length of X and Y. Thus even conditioned on the fixing of these random variables, X and Y still have plenty of entropy, which implies that the output of the extractor is still ϵ_2-close to uniform. Thus, as long as ϵ_2 is small, the output of the extractor is roughly independent of the event $\tilde{\alpha} \neq \tilde{\alpha}'$. Therefore, conditioned on any given output of the extractor, the event $\tilde{\alpha} \neq \tilde{\alpha}'$ still happens with probability roughly $1-\epsilon_1$ and we won't be paying a price of $2^m\epsilon_1$ here. Once this event happens, the correlation breaker ensures that the extractor is non-malleable with error ϵ_2, and we can use a similar argument as in [18] to get a non-malleable code with error roughly $2^m\epsilon_2$. Thus the total error of the non-malleable code is roughly $\epsilon_1 + 2^m\epsilon_2$. Now, we just need $m < \log(1/\epsilon_2)$.

We can now play with the two parameters ϵ_1, ϵ_2. The advice length L is $\Omega(\log(1/\epsilon_1))$ and we need to supply entropy $O(\frac{\log L}{\log\log L}\log(1/\epsilon_2))$ by using our improved correlation breaker. If we can achieve $L = \Theta(\log(1/\epsilon_1))$ then one can see that if we choose ϵ_1 to be any constant, then we can set $\epsilon_2 = 2^{-\Omega(n)}$ and also $m = \Omega(n)$, thus we get a constant rate non-malleable code. If we set $\epsilon_1 = 2^{-\mathsf{polylog}(n)}$ then we can set $\epsilon_2 = 2^{-\Omega(\frac{n\log\log\log n}{\log\log n})}$ and thus we get rate $\Omega(\frac{\log\log\log n}{\log\log n})$.

A technical issue here is how to achieve $L = \Theta(\log(1/\epsilon_1))$ for any ϵ_1. In [12, 54], the advice is obtained by using some random seed R to sample from an asymptotically good encoding of X, Y, and concatenating the sampled symbols with R. This puts a lower bound of $\log n$ on L, since we need at least this number of bits to sample from a string of length n. However this is not good enough to achieve constant rate. Our idea around this is to use repeated sampling. To illustrate the idea, suppose for example that we have obtained an advice V such that $V \neq V'$ with probability $1 - 1/\mathsf{poly}(n)$ and V has length $O(\log n)$. We now use another piece of independent random bits R_1 of length $O(\log\log n)$ to sample $O(\log\log n)$ bits from an asymptotically good encoding of V, and obtain a new advice V_1 by concatenating R_1 with the sample bits. This ensures that $V_1 \neq V_1'$ happens with probability $1 - 1/\mathsf{polylog}(n)$ conditioned on $V \neq V'$, and the length of V_1 is now $O(\log\log n)$. We repeat this process until we get the desired error ϵ_1 (e.g., a constant) and the advice length is now $L = \Theta(\log(1/\epsilon_1))$. Note that the total error is still $O(\epsilon_1)$, the total number of random bits needed is small, and the process terminates in roughly $\log^* n$ steps. To prepare the independent random bits used in repeated sampling, we first take a small slice of X and Y and do an alternating extraction with roughly $\log^* n$ steps, which guarantees the bits used for sampling in later steps are independent of the previous ones and their tampered versions. Finally, some extra work are needed here to take care of the issue of fixed points, which is more subtle than [18] since now we are treating the two errors ϵ_1 and ϵ_2 separately.

Organization. The rest of the paper is organized as follows. We give some preliminaries in Section 2, and define alternating extraction in Section 3. We present independence preserving mergers in Section 4, correlation breakers in Section 5, non-malleable extractors in Section 6, and non-malleable codes in Section 7. Finally we conclude with some open problems in Section 8.

2 Preliminaries

We often use capital letters for random variables and corresponding small letters for their instantiations. Let $|S|$ denote the cardinality of the set S. For ℓ a positive integer, U_ℓ denotes the uniform distribution on $\{0,1\}^\ell$. When used as a component in a vector, each U_ℓ is assumed independent of the other components. When we have adversarial tampering, we use letters with prime to denote the tampered version of random variables. All logarithms are to the base 2.

2.1 Probability Distributions

▶ **Definition 16** (statistical distance). *Let W and Z be two distributions on a set S. Their* statistical distance *(variation distance) is*

$$\Delta(W,Z) =: \max_{T\subseteq S}(|W(T) - Z(T)|) = \frac{1}{2}\sum_{s\in S}|W(s) - Z(s)|.$$

We say W is ε-close to Z, denoted $W \approx_\varepsilon Z$, if $\Delta(W,Z) \le \varepsilon$. For a distribution D on a set S and a function $h : S \to T$, let $h(D)$ denote the distribution on T induced by choosing x according to D and outputting $h(x)$.

▶ **Lemma 17.** *For any function α and two random variables A, B, we have $\Delta(\alpha(A), \alpha(B)) \le \Delta(A,B)$.*

2.2 Average Conditional Min Entropy

▶ **Definition 18.** *The* average conditional min-entropy *is defined as*

$$\begin{aligned}\widetilde{H}_\infty(X|W) &= -\log\left(\mathrm{E}_{w\leftarrow W}\left[\max_x \Pr[X=x|W=w]\right]\right)\\ &= -\log\left(\mathrm{E}_{w\leftarrow W}\left[2^{-H_\infty(X|W=w)}\right]\right).\end{aligned}$$

▶ **Lemma 19** ([28])**.** *For any $s>0$, $\Pr_{w\leftarrow W}[H_\infty(X|W=w) \ge \widetilde{H}_\infty(X|W) - s] \ge 1-2^{-s}$.*

▶ **Lemma 20** ([28])**.** *If a random variable B has at most 2^ℓ possible values, then $\widetilde{H}_\infty(A|B) \ge H_\infty(A) - \ell$.*

2.3 Prerequisites from Previous Work

Sometimes it is convenient to talk about average case seeded extractors, where the source X has average conditional min-entropy $\widetilde{H}_\infty(X|Z) \ge k$ and the output of the extractor should be uniform given Z as well. The following lemma is proved in [28].

▶ **Lemma 21** ([28])**.** *For any $\delta>0$, if Ext is a (k,ϵ) extractor then it is also a $(k+\log(1/\delta), \epsilon+\delta)$ average case extractor.*

For a strong seeded extractor with optimal parameters, we use the following extractor constructed in [41].

▶ **Theorem 22** ([41])**.** *For every constant $\alpha>0$, there exists a constant $\beta>0$ such that for all positive integers n,k and any $\epsilon > 2^{-\beta k}$, there is an explicit construction of a strong (k,ϵ)-extractor $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $d = O(\log n + \log(1/\epsilon))$ and $m \ge (1-\alpha)k$. The same statement also holds for a strong average case extractor.*

▶ **Theorem 23** ([19])**.** *For every $0 < m < n$ there is an explicit two-source extractor $\mathsf{IP} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^m$ based on the inner product function, such that if X, Y are two independent (n,k_1) and (n,k_2) sources respectively, then*

$$(\mathsf{IP}(X,Y), X) \approx_\epsilon (U_m, X) \text{ and } (\mathsf{IP}(X,Y), Y) \approx_\epsilon (U_m, Y),$$

where $\epsilon = 2^{-\frac{k_1+k_2-n-m-1}{2}}$.

The following standard lemma about conditional min-entropy is implicit in [60] and explicit in [56].

▶ **Lemma 24** ([56])**.** *Let X and Y be random variables and let $\mathcal{Y}$ denote the range of Y. Then for all $\epsilon>0$, one has*

$$\Pr_Y\left[H_\infty(X|Y=y) \ge H_\infty(X) - \log|\mathcal{Y}| - \log\left(\frac{1}{\epsilon}\right)\right] \ge 1-\epsilon.$$

We also need the following lemma.

▶ **Lemma 25.** *[49] Let (X,Y) be a joint distribution such that X has range $\mathcal{X}$ and Y has range $\mathcal{Y}$. Assume that there is another random variable X' with the same range as X such that $|X-X'| = \epsilon$. Then there exists a joint distribution (X',Y) such that $|(X,Y)-(X',Y)| = \epsilon$.*

3 Alternating Extraction

Our constructions use the following alternating extraction protocol as a key ingredient. Alternating extraction was first introduced in [35], and has now become an important tool in constructions related to extractors.

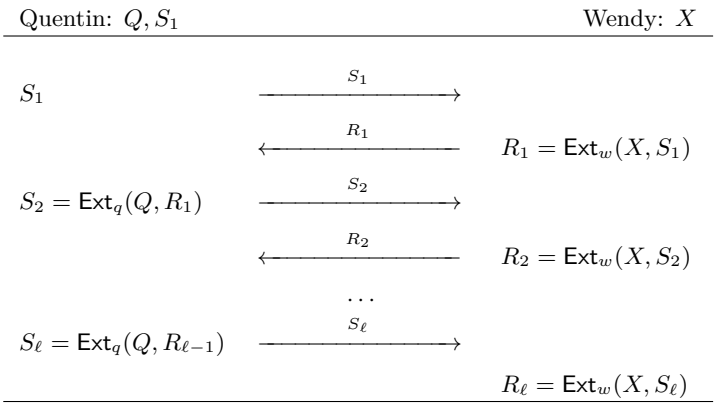


Figure 1 Alternating Extraction.

▶ **Definition 26** (Alternating Extraction). *Assume that we have two parties, Quentin and Wendy. Quentin has a source Q, Wendy has a source W. Also assume that Quentin has a uniform random seed S_1 (which may be correlated with Q). Suppose that (Q, S_1) is kept secret from Wendy and W is kept secret from Quentin. Let Ext_q, Ext_w be strong seeded extractors with optimal parameters, such as that in Theorem 22. Let r, s be two integer parameters for the protocol. For some integer parameter $\ell > 0$, the* alternating extraction protocol *is an interactive process between Quentin and Wendy that runs in ℓ steps.*

In the first step, Quentin sends S_1 to Wendy, Wendy computes $R_1 = \mathsf{Ext}_w(W, S_1)$. She sends R_1 to Quentin and Quentin computes $S_2 = \mathsf{Ext}_q(Q, R_1)$. In this step R_1, S_2 each outputs r and s bits respectively. In each subsequent step i, Quentin sends S_i to Wendy, Wendy computes $R_i = \mathsf{Ext}_w(W, S_i)$. She replies R_i to Quentin and Quentin computes $S_{i+1} = \mathsf{Ext}_q(Q, R_i)$. In step i, R_i, S_{i+1} each outputs r and s bits respectively. Therefore, this process produces the following sequence:

$$S_1, R_1 = \mathsf{Ext}_w(W, S_1), S_2 = \mathsf{Ext}_q(Q, R_1), \cdots,$$
$$S_\ell = \mathsf{Ext}_q(Q, R_{\ell-1}), R_\ell = \mathsf{Ext}_w(W, S_\ell).$$

The output of an alternating extraction protocol is often described as a *look-ahead extractor*, defined as follows. Let $Y = (Q, S_1)$ be a seed, the look-ahead extractor is defined as

$$\mathsf{laExt}(W, Y) = \mathsf{laExt}(W, (Q, S_1)) =: R_1, \cdots, R_\ell.$$

The following lemma is a special case of Lemma 6.5 in [12].

▶ **Lemma 27.** *Let W be an (n_w, k_w)-source and W' be a random variable on $\{0,1\}^{n_w}$ that is arbitrarily correlated with W. Let $Y = (Q, S_1)$ such that Q is a (n_q, k_q)-source, S_1 is a uniform string on s bits, and $Y' = (Q', S_1')$ be a random variable arbitrarily correlated with Y, where Q' and S_1' are random variables on n_q bits and s bits respectively. Let $\mathsf{Ext}_q, \mathsf{Ext}_w$ be strong seeded extractors that extract s and r bits from sources with min-entropy k with error ϵ and seed length $d \leq min\{r, s\}$. Suppose (Y, Y') is independent of (W, W'), $k_q \geqslant k + 2(\ell - 1)s + 2\log(\frac{1}{\epsilon})$,*

and $k_w \geqslant k + 2(\ell-1)r + 2\log(\frac{1}{\epsilon})$. *Let* laExt *be the look-ahead extractor defined above using* $\mathsf{Ext}_q, \mathsf{Ext}_w$, *and* $(R_1, \cdots, R_\ell) = \mathsf{laExt}(W, Y)$, $(R'_1, \cdots, R'_\ell) = \mathsf{laExt}(W', Y')$. *Then for any* $0 \leq j \leq \ell - 1$, *we have*

$$\begin{aligned}&(Y, Y', \{R_1, R'_1, \cdots, R_j, R'_j\}, R_{j+1})\\ \approx_{\epsilon_1} &(Y, Y', \{R_1, R'_1, \cdots, R_j, R'_j\}, U_r),\end{aligned}$$

where $\epsilon_1 = O(\ell\epsilon)$.

4 Non-Malleable Independence Preserving Merger

We now describe the notion of *non-malleable independence preserving merger*, introduced in [13] based on the notion of independence preserving merger introduced in [26].

▶ **Definition 28.** *A* (L, d', ε)-NIPM : $\{0,1\}^{Lm} \times \{0,1\}^d \to \{0,1\}^{m_1}$ *satisfies the following property. Suppose*

- $\mathbf{X}, \mathbf{X}'$ *are random variables, each supported on boolean* $L \times m$ *matrices s.t for any* $i \in [L]$, $\mathbf{X}_i = U_m$,
- $\{\mathbf{Y}, \mathbf{Y}'\}$ *is independent of* $\{\mathbf{X}, \mathbf{X}'\}$, *s.t* $\mathbf{Y}, \mathbf{Y}'$ *are each supported on* $\{0,1\}^d$ *and* $H_\infty(\mathbf{Y}) \geqslant d'$,
- *there exists an* $h \in [L]$ *such that* $(\mathbf{X}_h, \mathbf{X}'_h) = (U_m, \mathbf{X}'_h)$,

then

$$\begin{aligned}&|(L, d', \varepsilon)\text{-NIPM}(\mathbf{X}, \mathbf{Y}), (L, d', \varepsilon)\text{-NIPM}(\mathbf{X}', \mathbf{Y}')\\ &- U_{m_1}, (L, d', \varepsilon)\text{-NIPM}(\mathbf{X}', \mathbf{Y}')| \leqslant \epsilon.\end{aligned}$$

We have the following construction and theorem.

***L*-Alternating Extraction.** We extend the previous alternating extraction protocol by letting Quentin have access to L sources $Q_1, \ldots, Q_L$ (instead of just Q) which have the same length. Now in the i'th round of the protocol, he uses Q_i to produce the r.v $S_i = \mathsf{Ext}_q(Q_i, R_i)$. More formally, the following sequence of r.v's is generated: $S_1, R_1 = \mathsf{Ext}_w(W, S_1), S_2 = \mathsf{Ext}_q(Q_2, R_1), \ldots, R_{L-1} = \mathsf{Ext}_w(W, S_{\ell-1}), S_L = \mathsf{Ext}_q(Q_L, R_{L-1})$.

The NIPM is now constructed as follows. Let S_1 be a slice of $\mathbf{X}_1$ with length $O(\log(d/\varepsilon))$, then run the L-alternating extraction described above with $(Q_1, \ldots, Q_L) = (\mathbf{X}_1, \ldots, \mathbf{X}_L)$ and $W = \mathbf{Y}$. Finally output S_L.

▶ **Theorem 29** ([13]). *There exists a constant* $c > 0$ *such that for all integers* $m, d, d', L > 0$ *and any* $\epsilon > 0$, *with* $m \geqslant 4cL\log(d/\epsilon)$, $d' \geqslant 4cL\log(m/\epsilon)$, *the above construction* NIPM : $(\{0,1\}^m)^\ell \times \{0,1\}^d \to \{0,1\}^{m_1}$ *has output length* $m_1 \geq 0.2m$, *such that if the following conditions hold:*

- $\mathbf{X}, \mathbf{X}'$ *are random variables, each supported on boolean* $L \times m$ *matrices s.t for any* $i \in [L]$, $\mathbf{X}_i = U_m$,
- $\{\mathbf{Y}, \mathbf{Y}'\}$ *is independent of* $\{\mathbf{X}, \mathbf{X}'\}$, *s.t* $\mathbf{Y}, \mathbf{Y}'$ *are each supported on* $\{0,1\}^d$ *and* $H_\infty(\mathbf{Y}) \geqslant d'$,
- *there exists an* $h \in [L]$ *such that* $(\mathbf{X}_h, \mathbf{X}'_h) = (U_m, \mathbf{X}'_h)$,

then

$$|\text{NIPM}(\mathbf{X}, \mathbf{Y}), \text{NIPM}(\mathbf{X}', \mathbf{Y}'), \mathbf{Y}, \mathbf{Y}' - U_{m_1}, \text{NIPM}(\mathbf{X}', \mathbf{Y}'), \mathbf{Y}, \mathbf{Y}'| \leqslant L\epsilon.$$

It is sometimes more convenient to consider NIPMs which use an additional source X in the computation. We generalize the above definition as follows.

▶ **Definition 30.** *A* (L, d, d', ε)-NIPM : $\{0,1\}^{Lm} \times \{0,1\}^d \times \{0,1\}^{d'} \to \{0,1\}^{m_1}$ *satisfies the following property. Suppose*

- V, V' *are random variables, each supported on boolean* $L \times m$ *matrices s.t for any* $i \in [L]$, $V_i = U_m$,
- *there exists an* $h \in [L]$ *such that* $(V_h, V'_h) = (U_m, V'_h)$,
- $\mathbf{X}, \mathbf{X}'$ *are random variables, each supported on* d *bits, such that* $\mathbf{X}$ *is uniform conditioned on* (V, V'),
- $(\mathbf{Y}, \mathbf{Y}')$ *is independent of* $(V, V', \mathbf{X}, \mathbf{X}')$, *s.t* $\mathbf{Y}, \mathbf{Y}'$ *are each supported on* $\{0,1\}^{d'}$ *and* $\mathbf{Y}$ *is uniform,*

If the function is an NIPM *that is strong in* $\mathbf{Y}$ *then*

$$|(L, d, d', \varepsilon)\text{-NIPM}(V, \mathbf{X}, \mathbf{Y}), (L, d, d', \varepsilon)\text{-NIPM}(V', \mathbf{X}', \mathbf{Y}'), \mathbf{Y}, \mathbf{Y}' - U_{m_1}, (L, d, d', \varepsilon)\text{-NIPM}(V', \mathbf{X}', \mathbf{Y}'), \mathbf{Y}, \mathbf{Y}'| \leqslant \epsilon.$$

If the function is an NIPM *that is strong in* $\mathbf{X}$ *then*

$$|(L, d, d', \varepsilon)\text{-NIPM}(V, \mathbf{X}, \mathbf{Y}), (L, d, d', \varepsilon)\text{-NIPM}(V', \mathbf{X}', \mathbf{Y}'), \mathbf{X}, \mathbf{X}' - U_{m_1}, (L, d, d', \varepsilon)\text{-NIPM}(V', \mathbf{X}', \mathbf{Y}'), \mathbf{X}, \mathbf{X}'| \leqslant \epsilon.$$

We will now use the above construction to give another NIPM, which recycles the entropy. Specifically, we have the following construction.

▶ **Construction 31.** *Asymmetric* NIPM.

Inputs:

- $L, m, n, d \in \mathcal{N}$ *and an error parameter* $\epsilon > 0$ *such that* $m \geq c \log(d/\epsilon)$ *and* $d \geq c \log(n/\epsilon)$ *for some constant* $c > 1$.
- *A random variable* V *supported on a boolean* $L \times m$ *matrix.*
- *An* $(n, 6m)$ *source* $\mathbf{X}$.
- *Random variables* $\mathbf{Y}_1, \cdots, \mathbf{Y}_\ell$ *where* $\ell = \log L$ *and each* $\mathbf{Y}_i$ *is supported on* $\{0,1\}^d$.

Output: a random variable $\mathbf{W} \in \{0,1\}^m$.

Let $V^0 = V$. *For* $i = 1$ *to* $\log L$ *do the following.*

1. *Take a slice* $\mathbf{Y}_i^1$ *of* $\mathbf{Y}_i$ *with length* $d/3$. *Merge every two rows of* V^{i-1}, *using* $\mathbf{Y}_i^1$ *and the* NIPM *from Theorem 29. That is, for every* $j \leq t/2$ *where* t *is the current number of rows in* V^{i-1} *(initially* $t = L$*), compute* $\overline{V_j^{i-1}} = \text{NIPM}((V_{2j-1}^{i-1}, V_{2j}^{i-1}), \mathbf{Y}_i^1)$.
2. *For every* $j \leq t/2$, *compute* $\overline{\mathbf{Y}_{ij}} = \mathsf{Ext}_1(\mathbf{Y}_i, \overline{V_j^{i-1}})$, *where* Ext_1 *is the extractor in Theorem 22 and output* $d/4$ *bits.*
3. *For every* $i \leq t/2$, *compute* $\widetilde{V_j^{i-1}} = \mathsf{Ext}_2(\mathbf{X}, \overline{\mathbf{Y}_{ij}})$, *where* Ext_2 *is the extractor in Theorem 22 and output* m *bits.*
4. *Let* V^i *with the concatenation of* $\widetilde{V_j^{i-1}}, j = 1, \cdots, t/2$. *Note that the number of rows in* V^i *has decreased by a factor of* 2.

Finally output $\mathbf{W} = V^{\log L}$.

▶ **Lemma 32.** *There is a constant* $c > 1$ *such that suppose we have the following random variables:*

- V, V', *each supported on a boolean* $L \times m$ *matrix s.t for any* $i \in [L]$, $V_i = U_m$. *In addition, there exists an* $h \in [L]$ *such that* $(V_h, V'_h) = (U_m, V'_h)$.
- $\mathbf{X}, \mathbf{X}'$ *where* $\mathbf{X}$ *is an* $(n, 6m)$ *source.*

- *Random variables* $(\mathbf{Y}_1, \mathbf{Y}'_1), \cdots, (\mathbf{Y}_\ell, \mathbf{Y}'_\ell)$ *obtained from* $\mathbf{Y}, \mathbf{Y}'$ *deterministically, where* $\ell = \log L$*. These random variables satisfy the following look-ahead condition:* $\forall j < \ell$*, we have*

$$(\mathbf{Y}_j, \mathbf{Y}_1, \mathbf{Y}'_1, \cdots, \mathbf{Y}_{j-1}, \mathbf{Y}'_{j-1}) = (U_d, \mathbf{Y}_1, \mathbf{Y}'_1, \cdots, \mathbf{Y}_{j-1}, \mathbf{Y}'_{j-1}).$$

In addition, $(V, V', \mathbf{X}, \mathbf{X}')$ *is independent of* $(\mathbf{Y}, \mathbf{Y}')$*.*

Let $\mathbf{W}$ *be the output of the* NIPM *on* $(V, \mathbf{X}, \mathbf{Y}_1, \cdots, \mathbf{Y}_\ell)$ *and* $\mathbf{W}'$ *be the output of the* NIPM *on* $(V', \mathbf{X}', \mathbf{Y}'_1, \cdots, \mathbf{Y}'_\ell)$*. Then*

$$(\mathbf{W}, \mathbf{W}', \mathbf{Y}, \mathbf{Y}') \approx_{O(L\epsilon)} (U_m, \mathbf{W}', \mathbf{Y}, \mathbf{Y}').$$

Proof. We use induction to show the following claim.

▷ Claim 33. For every $0 \le i \le \ell = \log L$, the following holds after step i.

- V^i, V'^i are each supported on boolean $(t = L/2^i) \times m$ matrices s.t for any $j \in [t]$, $(V^i_j, \mathbf{Y}, \mathbf{Y}') \approx_{\epsilon_j} (U_m, \mathbf{Y}, \mathbf{Y}')$. In addition, there exists an $h \in [t]$ such that $(V^i_h, V'^i_h, \mathbf{Y}, \mathbf{Y}') \approx_{\epsilon_i} (U_m, V'^i_h, \mathbf{Y}, \mathbf{Y}')$. Here ϵ_i is the error after step i which satisfies that $\epsilon_0 = 0$ and $\epsilon_{i+1} \le 2\epsilon_i + 4\epsilon$.
- Conditioned on the fixing of $\mathbf{Y}_1, \mathbf{Y}'_1, \cdots, \mathbf{Y}_j, \mathbf{Y}'_j$, each of V^i and V'^i is a deterministic function of $V, V', \mathbf{X}, \mathbf{X}'$.

For the base case of $i = 0$, the claim clearly holds. Now assume that the claim holds for i, we show that it holds for $i + 1$.

We first fix $\mathbf{Y}_1, \mathbf{Y}'_1, \cdots, \mathbf{Y}_i, \mathbf{Y}'_i$. By the induction hypothesis, conditioned on the fixing of these random variables, each of V^i and V'^i is a deterministic function of $V, V', \mathbf{X}, \mathbf{X}'$, and thus independent of $(\mathbf{Y}_{i+1}, \mathbf{Y}'_{i+1})$. We only consider the row $h \in [t]$ such that $(V_h, V'_h) \approx_{4 \cdot 2^i \epsilon} (U_m, V'_h)$, since the analysis for the rest of the rows are similar and simpler.

First we ignore the error ϵ_i. By Theorem 29, and note that we are merging every two rows at one step, we can choose a suitable constant $c > 1$ in the construction such that

$$(\overline{V^i_{h'}}, \overline{V'^i_{h'}}, \mathbf{Y}^1_{i+1}, \mathbf{Y}'^1_{i+1}) \approx_{2\epsilon} (U_{m_1}, \overline{V'^i_{h'}}, \mathbf{Y}^1_{i+1}, \mathbf{Y}'^1_{i+1}),$$

where $h' = \lceil \frac{h}{2} \rceil$ and $m_1 = 0.2m$. We now fix $(\mathbf{Y}^1_{i+1}, \mathbf{Y}'^1_{i+1})$. Note that conditioned on the fixing, $\mathbf{Y}_{i+1}$ still has average conditional min-entropy at least $d - d/3 = 2d/3$ and is independent of $(\overline{V^i_{h'}}, \overline{V'^i_{h'}})$. Now we can first fix $\overline{V'^i_{h'}}$ and then $\overline{\mathbf{Y}'_{ih'}}$. Note that conditioned on this fixing, $\overline{V^i_{h'}}$ is still (close to) uniform and the average conditional min-entropy of $\mathbf{Y}_{i+1}$ is at least $2d/3 - d/4 > d/3$. Thus as long as c is large enough, by Theorem 22 we have that

$$(\overline{\mathbf{Y}_{ih'}}, \overline{V^i_{h'}}) \approx_\epsilon (U_{d/4}, \overline{V^i_{h'}}).$$

We now further fix $\overline{V^i_{h'}}$. Note that conditioned on this fixing, $\overline{\mathbf{Y}_{ih'}}$ is still (close to) uniform. Moreover conditioned on all the random variables we have fixed, $\overline{\mathbf{Y}}_{ih'}$ is a deterministic function of $\mathbf{Y}_1, \mathbf{Y}'_1, \cdots, \mathbf{Y}_{i+1}, \mathbf{Y}'_{i+1}$ and thus independent of $\mathbf{X}, \mathbf{X}'$. Also conditioned on all the random variables we have fixed, the average conditional min-entropy of $\mathbf{X}$ is at least $6m - 2m_1 > 5m$.

We can now further fix $\widetilde{V'^i_{h'}}$, which is a deterministic function of $\mathbf{X}'$. Conditioned on this fixing the independence of random variables still holds, while the average conditional min-entropy of $\mathbf{X}$ is at least $5m - m = 4m$. Therefore by Theorem 22 we have that

$$(\widetilde{V^i_{h'}}, \overline{\mathbf{Y}_{ih'}}) \approx_\epsilon (U_m, \overline{\mathbf{Y}_{ih'}}).$$

Since we have already fixed $\overline{\mathbf{Y}'_{ih'}}$ and $\widetilde{V'^i_{h'}}$, and note that conditioned on this fixing, $(\mathbf{Y}, \mathbf{Y}')$ are independent of $\widetilde{V^i_{h'}}$ which is a deterministic function of $\mathbf{X}$, we also have that

$$(\widetilde{V^i_{h'}}, \widetilde{\mathbf{X}'_{h'}}, \mathbf{Y}, \mathbf{Y}') \approx_\epsilon (U_m, \widetilde{\mathbf{X}'_{h'}}, \mathbf{Y}, \mathbf{Y}').$$

Adding back all the errors we get that there exists an $h' \in [t]$ such that

$$(\widetilde{\mathbf{X}_{h'}}, \widetilde{V'^i_{h'}}, \mathbf{Y}, \mathbf{Y}') \approx_{\epsilon_{i+1}} (U_m, \widetilde{V'^i_{h'}}, \mathbf{Y}, \mathbf{Y}'),$$

where $\epsilon_{i+1} \leq 2\epsilon_i + 4\epsilon$. Furthermore, it is clear that conditioned on the fixing of $\mathbf{Y}_1, \mathbf{Y}'_1, \cdots, \mathbf{Y}_{i+1}, \mathbf{Y}'_{i+1}$, each of V^{i+1} and V'^{i+1} is a deterministic function of $V, V', \mathbf{X}, \mathbf{X}'$.

We can now estimate the final error to be $\epsilon_\ell \leq 4(\sum_{i=1}^{\ell} 2^i \epsilon) = O(L\epsilon)$. Finally, when the number or rows in V^i decreases to 1 after step ℓ, the output $\mathbf{W} = V^{\log L}$ satisfies the conclusion of the lemma. ◀

We will now construct another NIPM. First we need the following lemma.

▶ **Lemma 34.** *For any constant $a \in \mathcal{N}$, any $\ell, s \in \mathcal{N}$ and any $\epsilon > 0$ there exists an explicit function $\mathsf{Conv}_a : \{0,1\}^n \times \{0,1\}^{a \cdot d} \to \{0,1\}^{\ell \cdot s}$ with $d = O(\log(n/\epsilon))$ and $n = 2^{O(a \cdot \ell^{\frac{1}{a}})} \cdot s$ such that the following holds. Let (Y, Y') be two random variables each on n bits, and Y is uniform. Let $(X = (X_1, \cdots, X_a), X' = (X'_1, \cdots, X'_a))$ be random variables each on $a \cdot d$ bits, where each X_i and X'_i is on d bits. Further assume that (X, X') satisfies the following look-ahead property: $\forall i \in [a]$, we have*

$$(X_i, X_1, X'_1, \cdots, X_{i-1}, X'_{i-1}) = (U_d, X_1, X'_1, \cdots, X_{i-1}, X'_{i-1}).$$

Let $(W_1, \cdots, W_\ell) = \mathsf{Conv}_a(Y, X)$ and $(W'_1, \cdots, W'_\ell) = \mathsf{Conv}_a(Y', X')$. Then we have

$$(X, X', W_1, W'_1, \cdots, W_\ell, W'_\ell) \approx_{O(\ell\epsilon)} (X, X', U_s, W'_1, \cdots, U_s, W'_\ell),$$

where each U_s is independent of previous random variables but may depend on later random variables.

Proof. We will prove the lemma by induction on a. For the base case $a = 1$, consider the following construction. For $j = 1, \cdots, \ell$, let Y_j be a slice of Y with length $(2^j - 1) \cdot 2s$ (this is possible since the total entropy required is at most $2^\ell \cdot 2s$), and compute $W_j = \mathsf{Ext}(Y_j, X_1)$. Note that for any $j \in [\ell]$, conditioned on the fixing of $Y_1, Y'_1, \cdots, Y_{j-1}, Y'_{j-1}$, the average conditional min-entropy of Y_j is at least $(2^j - 1) \cdot 2s - 2(2^{j-1} - 1) \cdot 2s = 2s$. Thus by Theorem 22 we have that

$$(W_j, Y_1, Y'_1, \cdots, Y_{j-1}, Y'_{j-1}, X, X') \approx_\epsilon (U_s, Y_1, Y'_1, \cdots, Y_{j-1}, Y'_{j-1}, X, X').$$

Since $(W_1, W'_1, \cdots, W_{j-1}, W'_{j-1})$ is a deterministic function of $(Y_1, Y'_1, \cdots, Y_{j-1}, Y'_{j-1})$ and (X, X'), we also have that

$$(W_j, W_1, W'_1, \cdots, W_{j-1}, W'_{j-1}, X, X') \approx_\epsilon (U_s, W_1, W'_1, \cdots, W_{j-1}, W'_{j-1}, W, W').$$

By adding all the errors the statement of the lemma holds.

Now assume that the lemma holds for a, we will construct another function Conv_{a+1} for the case of $a + 1$. First choose a parameter $t \in \mathcal{N}$ to be decided later. For $j = 1, \cdots, \ell/t$, let Y_j be a slice of Y with length $(2^j - 1) \cdot 2m$, where m is the length of Y (i.e., n) for Conv_a when choosing $\ell = t$. Thus we have $m = 2^{O(a \cdot t^{\frac{1}{a}})} \cdot s$. Now, for every j we first

use X_1 to compute $\hat{W}_j = \mathsf{Ext}(Y_j, X_1)$ and output m bits, then compute $(\hat{W}_{1j}, \cdots, \hat{W}_{tj}) = \mathsf{Conv}_a(\hat{W}_j, X_2, \cdots, X_{a+1})$. The final outputs are obtained by combining all the $\{\hat{W}_{ij}\}$ in sequence.

Note that by the same argument as above, we have that

$$(X_1, X_1', \hat{W}_1, \hat{W'}_1, \cdots, \hat{W}_{\ell/t}, \hat{W'}_{\ell/t}) \approx_{O(\frac{\ell}{t}\epsilon)} (X_1, X_1', U_m, \hat{W'}_1, \cdots, U_m, \hat{W'}_{\ell/t}).$$

Now we can fix (X_1, X_1'). Note that conditioned on the fixing, $(\hat{W}_1, \hat{W'}_1, \cdots, \hat{W}_{\ell/t}, \hat{W'}_{\ell/t})$ is a deterministic function of (Y, Y'), thus independent of (X, X'). Now we can used the induction hypothesis to conclude that the statement holds for the case of $a+1$. Note that the total error is $O(\frac{\ell}{t}\epsilon) + \ell/t \cdot O(t\epsilon) = O(\ell\epsilon)$ since the part of $O(\frac{\ell}{t}\epsilon)$ decreases as a geometric sequence. Finally, the entropy requirement of Y is $(2^{\ell/t} - 1) \cdot 2m = (2^{\ell/t} - 1) \cdot 2 \cdot 2^{O(a \cdot t^{\frac{1}{a}})} \cdot s = 2^{l/t + O(a \cdot t^{\frac{1}{a}}) + 1} \cdot s$.

We now just need to choose a t to minimize this quantity. We can choose $t = \ell^{\frac{a}{a+1}}$ so that the entropy requirement of Y is $2^{O((a+1) \cdot \ell^{\frac{1}{a+1}})} \cdot s$. ◀

We now have the following construction.

▶ **Construction 35.** NIPM_x *(which is strong in* Y*) or* NIPM_y *(which is strong in* X*).*
Inputs:

- *An error parameter* $\epsilon > 0$ *and a constant* $a \in \mathcal{N}$*.*
- *A random variable* V *supported on a boolean* $L \times m$ *matrix.*
- *A uniform string* X *on* d_1 *bits.*
- *A uniform string* Y *on* d_2 *bits.*
- *Let* $d = c\log(max\{d_1, d_2\}/\epsilon)$ *for some constant* $c > 1$*.*

Output: NIPM_x *outputs a random variable* $\mathbf{W}_x \in \{0,1\}^m$*, and* NIPM_y *outputs* $\mathbf{W}_y \in \{0,1\}^d$*.*

1. *Let* $\ell = \log L$.[5] *Let* X_0 *be a slice of* X *with length* $4a \cdot d$*, and* Y_0 *be a slice of* Y *with length* $4a \cdot d$*. Use* X_0 *and* Y_0 *to run an alternating extraction protocol, and output* $(R_0, \cdots, R_a) = \mathsf{laExt}(X_0, Y_0)$ *where each* R_i *has* d *bits.*
2. *Compute* $Z = \mathsf{Ext}(Y, R_0)$ *and output* $d_2/2$ *bits, where* Ext *is the strong seeded extractor from Theorem 22.*
3. *For every* $i \in [L]$*, compute* $\overline{V_i} = \mathsf{Ext}(Y_0, V_i)$ *and output* d *bits. Then, compute* $\hat{V}_i = \mathsf{Ext}(X, \overline{V_i})$ *and output* m *bits.*
4. *Compute* $(Z_1, \cdots, Z_\ell) = \mathsf{Conv}_a(Z, R_1, \cdots, R_a)$ *where each* Z_i *has* d *bits.*
5. NIPM_x *outputs* $\mathbf{W}_x = \mathrm{NIPM}(\hat{V}, Z_1, \cdots, Z_\ell)$*, where* NIPM *is the merger in Construction 31 and Lemma 32.* NIPM_y *outputs* $\mathbf{W}_y = \mathsf{Ext}(Y, \mathbf{W}_x)$ *with* d *bits.*

We now have the following lemma.

▶ **Lemma 36.** *There exist a constant* $c > 1$ *such that for any* $\epsilon > 0$ *and any* $L, m, d_1, d_2, n \in \mathcal{N}$ *such that* $d \geq c(\log\max\{d_1, d_2\} + \log(1/\epsilon))$*,* $m \geq d$*,* $d_1 \geq 8a \cdot d + 6m$ *and* $d_2 \geq 8a \cdot d + c^{a \cdot \log^{\frac{1}{a}} L} \cdot d$*, the above construction gives an* $(L, d_1, d_2, O(L\epsilon))$-NIPM *that is either strong in* X *or strong in* Y*.*

[5] Without loss of generality we assume that L is a power of 2. Otherwise add 0 to the string until the length is a power of 2.

Proof. Note that Y_0 has min-entropy $4ad \geq 4d$, thus by Theorem 22 we have that for every $i \in [L]$,

$$(\overline{V_i}, V_i) \approx_\epsilon (U_d, V_i),$$

and there exists an $h \in [L]$ such that

$$(\overline{V_h}, \overline{V'_h}, V_h, V'_h) \approx_\epsilon (U_d, \overline{V'_h}, V_h, V'_h).$$

Note that conditioned on the fixing of (V, V'), we have that (X, X') and (Y, Y') are still independent, and furthermore $(\overline{V}, \overline{V'})$ is a deterministic function of (Y, Y'). Note that conditioned on the fixing of (X_0, X'_0), the average conditional min-entropy of X is at least $8a \cdot d + 6m - 2 \cdot 4a \cdot d = 6m$. Thus again by Theorem 22 we have that for every $i \in [L]$,

$$(\hat{V}_i, \overline{V_i}) \approx_\epsilon (U_d, \overline{V_i}),$$

and there exists an $h \in [L]$ such that

$$(\hat{V}_h, \hat{V'}_h, \overline{V_h}, \overline{V'_h}) \approx_\epsilon (U_d, \hat{V'}_h, \overline{V_h}, \overline{V'_h}).$$

Note that now conditioned on the fixing of $(\overline{V_h}, \overline{V'_h})$, we have that (X, X') and (Y, Y') are still independent, and furthermore $(\hat{V}_h, \hat{V'}_h)$ is a deterministic function of (X, X'). Thus we basically have that conditioned on the fixing of (X_0, X'_0, Y_0, Y'_0), $(\hat{V}, \hat{V'})$ is a deterministic function of (X, X') and they satisfy the property needed by an NIPM.

Now, by Lemma 27, we have that

$$(Y_0, Y'_0, R_0, R'_0, \cdots, R_a, R'_a) \approx_{O(a^2\epsilon)} (Y_0, Y'_0, U_d, R'_0, \cdots, U_d, R'_a).$$

Note that conditioned on the fixing of (Y_0, Y'_0), we have that (X, X') and (Y, Y') are still independent, and furthermore $(R_0, R'_0, \cdots, R_a, R'_a)$ is a deterministic function of (X, X'). Also the average conditional min-entropy of Y is at least $d_2 - 2 \cdot 4a \cdot d = c^{a \cdot \log^{\frac{1}{a}} L} \cdot d > 3d_2/4$ for a large enough constant c. Thus by Theorem 22 we have that

$$(Z, R_0) \approx_\epsilon (U_{d_2/2}, R_0).$$

We can now fix (R_0, R_0). Note that now (Z_0, Z'_0) is a deterministic function of (Y, Y'), and $d_2/2 > \frac{1}{2} c^{a \cdot \log^{\frac{1}{a}} L} \cdot d$. Note that now $(R_1, R'_1, \cdots, R_a, R'_a)$ still satisfies the look-ahead property. Thus as long as c is large enough, by Lemma 34 we have that

$$(Z_1, Z'_1, \cdots, Z_\ell, Z'_\ell, X_0, X'_0) \approx_{O(\ell\epsilon)} (U_d, W'_1, \cdots, U_d, W'_\ell, X_0, X'_0).$$

We can now fix (X_0, X'_0), and note that conditioned on this fixing $(Z_1, Z'_1, \cdots, Z_\ell, Z'_\ell)$ is a deterministic function of (Y, Y'). In summary, conditioned on the fixing of (X_0, X'_0, Y_0, Y'_0), we have that $(\hat{V}, \hat{V'})$and $(Z_1, Z'_1, \cdots, Z_\ell, Z'_\ell)$ satisfy the conditions required by Lemma 32. Therefore we can now apply that lemma to finish the proof. The total error is at most $O(L\epsilon) + O(a^2\epsilon) + O(\epsilon) + O(\ell\epsilon) = O(L\epsilon)$. ◀

The extreme case of the above construction gives the following NIPM.

▶ **Construction 37.** NIPM_x *(which is strong in* Y*) or* NIPM_y *(which is strong in* X*).*

Inputs:

- *An error parameter* $\epsilon > 0$.
- *A random variable* V *supported on a boolean* $L \times m$ *matrix.*
- *A uniform string* $\mathbf{X}$ *on* n *bits.*
- *A uniform string* $\mathbf{Y}$ *on* n' *bits.*

Output: NIPM_x *outputs a random variable* $\mathbf{W}_x \in \{0,1\}^m$*, and* NIPM_y *outputs* $\mathbf{W}_y \in \{0,1\}^{O(\log(n/\epsilon))}$.

1. *Let* $d_1 = c\log(n'/\epsilon)$ *and* $d_2 = c\log(n/\epsilon)$*. Take a slice* $\mathbf{X}_0$ *of* $\mathbf{X}$ *with length* $10\log\log L \cdot d_1$*, and a slice* $\mathbf{Y}_0$ *of* $\mathbf{Y}$ *with length* $10\log\log L \cdot d_2$.
2. *Use* $\mathbf{X}_0$ *and* $\mathbf{Y}_0$ *to do an alternating extraction protocol, and output* $(R_0, R_1, \cdots, R_t) = \mathsf{laExt}(\mathbf{X}_0, \mathbf{Y}_0)$ *where* $t = \log\log L$ *and each* R_i *has* $4d_1$ *bits, each* S_i *(used in the alternating extraction) has* d_2 *bits.*
3. *For each* $i \in [L]$*, compute* $\overline{\mathbf{Y}}_i = \mathsf{Ext}(\mathbf{Y}_0, V_i)$ *where each* $\overline{\mathbf{Y}}_i$ *outputs* d_2 *bits. Then compute* $\overline{V}_i = \mathsf{Ext}(\mathbf{X}, \overline{\mathbf{Y}}_i)$ *where each* $\overline{V}_i$ *outputs* m *bits. Here* Ext *is the strong seeded extractor from Theorem 22. Let* $\overline{V}$ *be the matrix whose* i*'th row is* $\overline{V}_i$.
4. *Let* $\mathbf{Y}_1^0 = \mathbf{Y}$*. For* $j = 0$ *to* $\log\log L$ *do the following. For* $h = 1$ *to* 2^j*, use* $\mathbf{Y}_h^j$ *and* R_j *to do an alternating extraction protocol, and output* $(S_{h1}^j, S_{h2}^j) = \mathsf{laExt}(\mathbf{Y}_h^j, R_j)$*, where each* S_{hi}^j *has* $(\frac{\log^{\log a} L}{a^{j-1}} - 1)d_2$ *bits. Note that altogether we get* 2^{j+1} *outputs and relabel them as* $\mathbf{Y}_1^{j+1}, \cdots, \mathbf{Y}_{2^{j+1}}^{j+1}$.
5. *After the previous step, we get* $2\log L$ *outputs. Let them be* $\mathbf{Y}_1, \cdots, \mathbf{Y}_{2\log L}$*, and output* $\mathbf{W}_x = \mathrm{NIPM}(\overline{V}, \mathbf{X}, \mathbf{Y}_1, \cdots, \mathbf{Y}_{2\log L})$ *with* m *bits. Let* $\mathbf{W}_y = \mathsf{Ext}(\mathbf{Y}, \mathbf{W}_x)$ *with* d_2 *bits.*

We now have the following lemma.

▶ **Lemma 38.** *There is a constant* $c > 1$ *such that suppose we have the following random variables and conditions:*

- V, V'*, each supported on a boolean* $L \times m$ *matrix s.t for any* $i \in [L]$*,* $V_i = U_m$*. In addition, there exists an* $h \in [L]$ *such that* $(V_h, V_h') = (U_m, V_h')$.
- $\mathbf{Y}, \mathbf{Y}'$*, each supported on* n' *bits, where* $\mathbf{Y}$ *is uniform.*
- $\mathbf{X}, \mathbf{X}'$*, each supported on* n *bits, where* $\mathbf{X}$ *is uniform. In addition,* $\mathbf{X}$ *is independent of* (V, V')*, and* $(V, V', \mathbf{X}, \mathbf{X}')$ *is independent of* $(\mathbf{Y}, \mathbf{Y}')$.
- $m \geq c\log(n'/\epsilon)$*,* $n \geq 20c\log\log L\log(n'/\epsilon) + 6m$ *and* $n' \geq 20c\log^{\log a} L\log(n/\epsilon)$.

Let $(\mathbf{W}_x, \mathbf{W}_y)$ *be the outputs of* $(\mathrm{NIPM}_x, \mathrm{NIPM}_y)$ *on* $(V, \mathbf{X}, \mathbf{Y})$ *and* $(\mathbf{W}_x', \mathbf{W}_y')$ *be the outputs of the* $(\mathrm{NIPM}_x, \mathrm{NIPM}_y)$ *on* $(V', \mathbf{X}', \mathbf{Y}')$*. Then*

$$(\mathbf{W}_x, \mathbf{W}_x', \mathbf{Y}, \mathbf{Y}') \approx_{O(L\epsilon)} (U_m, \mathbf{W}_x', \mathbf{Y}, \mathbf{Y}')$$

and

$$(\mathbf{W}_y, \mathbf{W}_y', V, V', \mathbf{X}, \mathbf{X}') \approx_{O(L\epsilon)} (U_{O(\log(n/\epsilon))}, \mathbf{W}_y', V, V', \mathbf{X}, \mathbf{X}').$$

Proof. First, since $(V, V', \mathbf{X}, \mathbf{X}')$ is independent of $(\mathbf{Y}, \mathbf{Y}')$, as long as c is large enough, by Theorem 22 we know that for any $i \in [L]$,

$$(\overline{\mathbf{Y}}_i, V) \approx_\epsilon (U_d, V).$$

In addition, suppose for some $h \in [L]$ we have that $(V_h, V_h') = (U_m, V_h')$, then we can first fix V_h' and then $\overline{\mathbf{Y}}_h$. Conditioned on this fixing V_h is still uniform, the average conditional min-entropy of $\mathbf{Y}_0$ is at least $10\log\log L \cdot d - d > 3d$ and V_h and Y_0 are still independent, thus by Theorem 22 we have that

$$(\overline{\mathbf{Y}}_h, \overline{\mathbf{Y}}_h', V, V') \approx_\epsilon (U_d, \overline{\mathbf{Y}}_h', V, V').$$

In other words, the random variables $\{(\overline{\mathbf{Y}}_i, \overline{\mathbf{Y}}_i')\}$ inherit the properties of $\{(V_i, V_i')\}$. We now ignore the errors since this adds at most $L\epsilon$ to the final error. Now we fix (V, V'). Note that conditioned on this fixing, the random variables $(\overline{\mathbf{Y}}_i, \overline{\mathbf{Y}}_i')$ are deterministic functions of $(\mathbf{Y}_0, \mathbf{Y}_0')$, and are thus independent of $(\mathbf{X}, \mathbf{X}')$. Furthermore, we have that conditioned on this

fixing, $\mathbf{X}$ is still uniform. In addition, even conditioned on the fixing of $(\mathbf{X}_0, \mathbf{X}'_0)$, the average conditional min-entropy of $\mathbf{X}$ is at least $20c \log\log L \log(n'/\epsilon) + 6m - 2 \cdot 10 \log\log L \cdot d_1 = 6m$. Thus by the same argument before we have that for any $i \in [L]$,

$$(\overline{V}_i, \mathbf{Y}_0, \mathbf{X}_0, \mathbf{X}'_0) \approx_\epsilon (U_m, \mathbf{Y}_0, \mathbf{X}_0, \mathbf{X}'_0),$$

and that there exists an $h \in [L]$ such that

$$(\overline{V}_h, \overline{V}'_h, \mathbf{Y}_0, \mathbf{Y}'_0, \mathbf{X}_0, \mathbf{X}'_0) \approx_\epsilon (U_m, \overline{V}'_h, \mathbf{Y}_0, \mathbf{Y}'_0, \mathbf{X}_0, \mathbf{X}'_0).$$

We will again ignore the error for now since this adds at most $L\epsilon$ to the final error. Next, by Lemma 27 we have that for any $0 \le j \le t-1$,

$$(R_{j+1}, (R_1, R'_1, \cdots, R_j, R'_j), \mathbf{Y}_0, \mathbf{Y}'_0) \approx_{O(t\epsilon)} (U_{4d_1}, (R_1, R'_1, \cdots, R_j, R'_j), \mathbf{Y}_0, \mathbf{Y}'_0).$$

Thus by a hybrid argument and the triangle inequality, we have that

$$(\mathbf{Y}_0, \mathbf{Y}'_0, R_1, R'_1, \cdots, R_t, R'_t) \approx_{O(t^2\epsilon)} (\mathbf{Y}_0, \mathbf{Y}'_0, U_{4d_1}, R'_1, \cdots, U_{4d_1}, R'_t),$$

where each U_{4d_1} is independent of all the previous random variables (but may depend on later random variables). From now on, we will proceed as if each R_j is uniform given $(\mathbf{Y}_0, \mathbf{Y}'_0, \{R_1, R'_1, \cdots, R_{j-1}, R'_{j-1}\})$, since this only adds $O(t^2\epsilon)$ to the final error.

Now we can fix $(\mathbf{Y}_0, \mathbf{Y}'_0)$. Note that conditioned on this fixing, $(\overline{V}, \overline{V}', R_1, R'_1, \cdots, R_t, R'_t)$ are deterministic functions of $(V, V', \mathbf{X}, \mathbf{X}')$, and thus independent of $(\mathbf{Y}, \mathbf{Y}')$. Also note that conditioned on this fixing, the average conditional min-entropy of $\mathbf{Y}$ is at least $20 \log^{\log a} L \cdot d_2 - 2 \cdot 10 \log\log L \cdot d_2 > a^2 \log^{\log a} L \cdot d_2$. We now prove the following claim.

▷ Claim 39. Let $\overline{R_j} = (R_1, \cdots, R_j)$. Suppose that at the beginning of the j'th iteration, we have that conditioned on the fixing of $\overline{R_{j-1}}$, the following holds.

1. , $(\mathbf{X}, \mathbf{X}')$ is independent of $(\mathbf{Y}, \mathbf{Y}')$, and $(\mathbf{Y}_1, \mathbf{Y}'_1, \cdots, \mathbf{Y}_{2^j}, \mathbf{Y}'_{2^j})$ is a deterministic function of $(\mathbf{Y}, \mathbf{Y}')$.
2. For every $h \in [2^j]$, the average conditional min-entropy of $\mathbf{Y}_h$ given $(\mathbf{Y}_1, \mathbf{Y}'_1, \cdots, \mathbf{Y}_{h-1}, \mathbf{Y}'_{h-1})$ is at least $(\frac{\log^{\log a} L}{a^{j-2}} - 1)d_2$.

Then at the end of the j'th iteration, the following holds.

1. Conditioned on the fixing of $\overline{R_j}$, $(\mathbf{X}, \mathbf{X}')$ is independent of $(\mathbf{Y}, \mathbf{Y}')$, and $(\mathbf{Y}_1, \mathbf{Y}'_1, \cdots, \mathbf{Y}_{2^{j+1}}, \mathbf{Y}'_{2^{j+1}})$ is a deterministic function of $(\mathbf{Y}, \mathbf{Y}')$.
2. For every $h \in [2^{j+1}]$,

$$(\mathbf{Y}_h, (\mathbf{Y}_1, \mathbf{Y}'_1, \cdots, \mathbf{Y}_{h-1}, \mathbf{Y}'_{h-1}), \overline{R_j}) \approx_\epsilon (U_{(\frac{\log^{\log a} L}{a^{j-1}} - 1)d_2}, (\mathbf{Y}_1, \mathbf{Y}'_1, \cdots, \mathbf{Y}_{h-1}, \mathbf{Y}'_{h-1}), \overline{R_j}).$$

Proof of the claim. First, since the computation in the j'th iteration only involves (R_j, R'_j) and $(\mathbf{Y}_1, \mathbf{Y}'_1, \cdots, \mathbf{Y}_{2^j}, \mathbf{Y}'_{2^j})$, and (R_j, R'_j) is a deterministic function of $(\mathbf{X}, \mathbf{X}')$ conditioned on the fixing of the previous random variables, we know that at the end of the j'th iteration, conditioned on the fixing of $(R_1, \cdots, R_j)$ we have that $(\mathbf{X}, \mathbf{X}')$ is independent of $(\mathbf{Y}, \mathbf{Y}')$, and $(\mathbf{Y}_1, \mathbf{Y}'_1, \cdots, \mathbf{Y}_{2^{j+1}}, \mathbf{Y}'_{2^{j+1}})$ is a deterministic function of $(\mathbf{Y}, \mathbf{Y}')$.

Next, we use $(Z_1, Z'_1, \cdots, Z_{2^{j+1}}, Z'_{2^{j+1}})$ to represent the outputs computed from (R_j, R'_j) and $(\mathbf{Y}_1, \mathbf{Y}'_1, \cdots, \mathbf{Y}_{2^j}, \mathbf{Y}'_{2^j})$, and assume that $2\ell - 1 \le h \le 2\ell$ for some ℓ, then Z_h is obtained from $\mathbf{Y}_\ell$. We can now first fix $(\mathbf{Y}_1, \mathbf{Y}'_1, \cdots, \mathbf{Y}_{\ell-1}, \mathbf{Y}'_{\ell-1})$, and conditioned on this fixing

$\mathbf{Y}_\ell$ has average conditional min-entropy at least $(\frac{\log^{\log a} L}{a^{j-2}} - 1)d_2$. Now by Lemma 27 we have that

$$(S_1^\ell, R_j, R'_j) \approx_\epsilon (U_{(\frac{\log^{\log a} L}{a^{j-1}}-1)d_2}, R_j, R'_j)$$

and

$$(S_2^\ell, S_1^\ell, S_1'^\ell, R_j, R'_j) \approx_\epsilon (U_{(\frac{\log^{\log a} L}{a^{j-1}}-1)d_2}, S_1^\ell, S_1'^\ell, R_j, R'_j),$$

since $(\frac{\log^{\log a} L}{a^{j-2}} - 1)d_2 \geq 2 \cdot (\frac{\log^{\log a} L}{a^{j-1}} - 1)d_2 + (1+\alpha)(\frac{\log^{\log a} L}{a^{j-1}} - 1)d_2 + d_2$ and $4d_1 \geq 2d_1 + 1.1d_1 + 0.9d_1$. Thus as long as the constant c is large enough one can make sure that $min\{d_2, 0.9d_1\} \geq 2\log(1/\epsilon)$, and we can extract $(\frac{\log^{\log a} L}{a^{j-1}} - 1)d_2$ bits from entropy $(1+\alpha)(\frac{\log^{\log a} L}{a^{j-1}} - 1)d_2$ and d_1 bits from entropy $1.1d_1$. Note that $(Z_1, Z'_1, \cdots, Z_{2\ell-2}, Z'_{2\ell-2})$ are computed from $(\mathbf{Y}_1, \mathbf{Y}'_1, \cdots, \mathbf{Y}_{\ell-1}, \mathbf{Y}'_{\ell-1})$ and (R_j, R'_j), and $(\mathbf{Y}_1, \mathbf{Y}'_1, \cdots, \mathbf{Y}_{\ell-1}, \mathbf{Y}'_{\ell-1})$ are already fixed. Thus the second part of the claim also holds. ◁

Now note that at the beginning of the first iteration, the condition of the claim holds. Thus if we ignore the errors, then we can apply the claim repeatedly until the end of the iteration. At this time for each $h \in [\log L]$ we have that $\mathbf{Y}_h$ has at least $(\frac{\log^{\log a} L}{a^{\log\log L-1}} - 1)d_2 > d_2$ bits. Furthermore

$$(\mathbf{Y}_h, (\mathbf{Y}_1, \mathbf{Y}'_1, \cdots, \mathbf{Y}_{h-1}, \mathbf{Y}'_{h-1}), \overline{R_t}) \approx (U, (\mathbf{Y}_1, \mathbf{Y}'_1, \cdots, \mathbf{Y}_{h-1}, \mathbf{Y}'_{h-1}), \overline{R_t}).$$

The total error so far is $O(L\epsilon) + O(t^2\epsilon) + \sum_{j=0}^{\log\log L} 2^j \cdot 2\epsilon = O(L\epsilon)$. Note that now conditioned on all the fixed random variables $(\mathbf{X}_0, \mathbf{X}'_0, \mathbf{Y}_0, \mathbf{Y}'_0, \overline{R_t})$ (note that $\overline{R_t}$ is a deterministic function of $(\mathbf{X}_0, \mathbf{X}'_0, \mathbf{Y}_0, \mathbf{Y}'_0)$, we have that $(V, V', \mathbf{Y}_1, \mathbf{Y}'_1, \cdots, \mathbf{Y}_{2\log L}, \mathbf{Y}'_{2\log L}, \mathbf{X}, \mathbf{X}')$ satisfies the conditions of the Lemma 32, since the average conditional min-entropy of X is at least $n - 20\log\log L \cdot d_1 \geq 6m$. Now we can apply Lemma 32 to show that

$$(\mathbf{W}_x, \mathbf{W}'_x, \mathbf{Y}, \mathbf{Y}') \approx (U_m, \mathbf{W}'_x, \mathbf{Y}, \mathbf{Y}'),$$

where the total error is $O(L\epsilon)+O(L\epsilon) = O(L\epsilon)$. Furthermore, note that conditioned on the fixing of $(\mathbf{Y}_1, \mathbf{Y}'_1, \cdots, \mathbf{Y}_{2\log L}, \mathbf{Y}'_{2\log L})$, we have that $(\mathbf{W}_x, \mathbf{W}'_x)$ is a deterministic function of $(V, V', \mathbf{X}, \mathbf{X}')$, and thus independent of $(\mathbf{Y}, \mathbf{Y}')$. Also note that $\mathbf{Y}$ has average conditional min-entropy at least $20c\log^{\log a} L\log(n/\epsilon) - 4\log L d_2 > 10d_2$. Thus by Theorem 22 we have that

$$(\mathbf{W}_y, \mathbf{W}'_y, \mathbf{W}_x, \mathbf{W}'_x) \approx (U_{d_2}, \mathbf{W}'_y, \mathbf{W}_x, \mathbf{W}'_x),$$

where the error is $O(L\epsilon)+O(\epsilon) = O(L\epsilon)$. Note that given $(\mathbf{W}_x, \mathbf{W}'_x)$, we have that $(\mathbf{W}_y, \mathbf{W}'_y)$ is a deterministic function of $(\mathbf{Y}, \mathbf{Y}')$. Thus we also have that

$$(\mathbf{W}_y, \mathbf{W}'_y, V, V', \mathbf{X}, \mathbf{X}') \approx_{O(L\epsilon)} (U_{d_2}, \mathbf{W}'_y, V, V', \mathbf{X}, \mathbf{X}').$$ ◀

5 Correlation Breaker with Advice

We now use our non-malleable independence preserving mergers to construct improved correlation breakers with advice. A correlation breaker uses independent randomness to break the correlations between several correlated random variables. The first correlation breaker appears implicitly in the author's work [49], and this object is strengthened and formally defined in [20]. A correlation breaker with advice additionally uses some string as an advice. This object was first introduced and used without its name in [12], and then explicitly defined in [22].

▶ **Definition 40** (Correlation breaker with advice). *A function*

$$\mathsf{AdvCB} : \{0,1\}^n \times \{0,1\}^d \times \{0,1\}^L \to \{0,1\}^m$$

is called a (k, k', ε)-correlation breaker with advice if the following holds. Let Y, Y' be d-bit random variables such that $H_\infty(Y) \geq k'$. Let X, X' be n-bit random variables with $H_\infty(X) \geq k$, such that (X, X') is independent of (Y, Y'). Then, for any pair of distinct L-bit strings α, α',

$$(\mathsf{AdvCB}(X, Y, \alpha), \mathsf{AdvCB}(X', Y', \alpha')) \approx_\varepsilon (U, \mathsf{AdvCB}(X', Y', \alpha')).$$

In addition, we say that AdvCB *is strong if*

$$\begin{aligned}&(\mathsf{AdvCB}(X, Y, \alpha), \mathsf{AdvCB}(X', Y', \alpha'), Y, Y')\\ \approx_\varepsilon &(U, \mathsf{AdvCB}(X', Y', \alpha'), Y, Y').\end{aligned}$$

Our construction needs the following flip-flop extraction scheme, which was constructed by Cohen [20] using alternating extraction, based on a previous similar construction of the author [49]. The flip-flop function can be viewed as a basic correlation breaker, which (informally) uses an independent source $\mathbf{X}$ to break the correlation between two r.v's $\mathbf{Y}$ and $\mathbf{Y}'$, given an advice bit.

▶ **Theorem 41** ([20, 12]). *There exists a constant c_{41} such that for all $n > 0$ and any $\epsilon > 0$, there exists an explicit function* flip-flop $: \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$, *$m = 0.4k$, satisfying the following: Let $\mathbf{X}$ be an (n, k)-source, and $\mathbf{X}'$ be a random variable on n bits arbitrarily correlated with $\mathbf{X}$. Let $\mathbf{Y}$ be an independent uniform seed on d bits, and $\mathbf{Y}'$ be a random variable on d bits arbitrarily correlated with $\mathbf{Y}$. Suppose $(\mathbf{X}, \mathbf{X}')$ is independent of $(\mathbf{Y}, \mathbf{Y}')$. If $k, d \geqslant C_{41} \log(n/\epsilon)$, then for any bit b,*

$$|\text{flip-flop}(\mathbf{X}, \mathbf{Y}, b), \mathbf{Y}, \mathbf{Y}' - U_m, \mathbf{Y}, \mathbf{Y}'| \leqslant \epsilon.$$

Furthermore, for any bits b, b' with $b \neq b'$,

$$\begin{aligned}&|\text{flip-flop}(\mathbf{X}, \mathbf{Y}, b), \text{flip-flop}(\mathbf{X}', \mathbf{Y}', b'), \mathbf{Y}, \mathbf{Y}'\\ &- U_m, \text{flip-flop}(\mathbf{X}', \mathbf{Y}', b'), \mathbf{Y}, \mathbf{Y}'| \leqslant \epsilon.\end{aligned}$$

5.1 Asymmetric correlation breaker

We will present correlation breakers that use general NIPMs. By plugging in various NIPMs this gives different correlation breakers.

▶ **Construction 42.** *Inputs:*

- *Let $\ell, m \in \mathcal{N}$ be two integers, $\epsilon > 0$ be an error parameter.*
- *X, Y, two independent sources on n bits and s bits respectively, with min-entropy at least $n - \ell$ and $s - \ell$.*
- *an advice string $\alpha \in \{0,1\}^L$.*
- *An $(L, d_1, d_2, O(L\epsilon))$-NIPM$_x$ that is strong in Y.*
- *Let* IP *be the two source extractor from Theorem 23.*

1. *Let $d' = O(\log(max\{n, s\}/\epsilon))$ be the seed length of the extractor from Theorem 22, and let $d = 8d'$. Let X^0 be a slice of X with length $d + 2\ell + 2\log(1/\epsilon)$, and Y^0 be a slice of Y with length $d + 2\ell + 2\log(1/\epsilon)$.*
2. *Compute $Z = \mathsf{IP}(X^0, Y^0)$ and output d bits.*

3. *Use X and Z to do an alternating extraction, and output two random variables $(X_0, X_1) = \mathsf{IaExt}(X, Z)$ where each X_i has $3m$ bits.*
4. *Use Y and Z to do an alternating extraction, and output two random variables $(Y_0, Y_1) = \mathsf{IaExt}(Y, Z)$ where each Y_i has $3d$ bits.*
5. *Use X_1, Y_1, α to obtain an $L \times m$ matrix V, where for any $i \in [L]$, $V_i = \text{flip-flop}(X_1, Y_1, \alpha_i)$ and outputs m bits.*
6. *Compute $\hat{X} = \mathsf{Ext}(X, Y_0)$ and output $n/2$ bits. Compute $\hat{Y} = \mathsf{Ext}(Y, X_0)$ and output $s/2$ bits. Here Ext is the strong seeded extractor from Theorem 22.*
7. *Output $\hat{V} = \text{NIPM}_x(V, \hat{X}, \hat{Y})$.*

We now have the following lemma.

▶ **Lemma 43.** *There exists a costant $c > 1$ such that the following holds. Suppose that there exists an $(L, d_1, d_2, O(L\epsilon))$-NIPM that is strong in Y which outputs m bits, then there exists an explicit $(n - \ell, s - \ell, O(L\epsilon))$ $\mathsf{AdvCB} : \{0,1\}^n \times \{0,1\}^s \times \{0,1\}^L \to \{0,1\}^m$ as long as $m \geq c \log(max\{n, s\}/\epsilon)$, $n \geq 20m + 2d_1 + 5\ell + 4\log(1/\epsilon)$ and $s \geq m + 2d_2 + 5\ell + 4\log(1/\epsilon)$.*

Proof. Throughout the proof we will use letters with prime to denote the corresponding random variables obtained from (X', Y', α'). First, notice that both X^0 and Y^0 have min-entropy at least $d + \ell + 2\log(1/\epsilon)$. Thus by Theorem 23 we have that

$$(Z, X^0) \approx_\epsilon (U_d, X^0)$$

and

$$(Z, Y^0) \approx_\epsilon (U_d, Y^0).$$

We now ignore the error ϵ. Note that conditioned on the fixing of (X^0, X'^0), (Z, Z') is a deterministic function of (Y^0, Y'^0), and thus independent of (X, X'). Moreover, the average conditional min-entropy of X given this fixing is at least $n - \ell - 2(d + 2\ell + 2\log(1/\epsilon)) \geq 10m$ as long as c is large enough. Thus by Lemma 27 (note that the extractor from Z side can use seed length d') we have that

$$(Y^0, Y'^0, X_0, X_0', X_1, X_1', Z, Z') \approx_{O(\epsilon)} (Y^0, Y'^0, U_{3m}, X_0', U_{d_1}, X_1', Z, Z'),$$

where each U_{3m} is uniform given the previous random variables, but may depend on later random variables. Similarly, note that conditioned on the fixing of (Y^0, Y'^0), (Z, Z') is a deterministic function of (X^0, X'^0), and thus independent of (Y, Y'). Moreover, the average conditional min-entropy of Y given this fixing is at least $s - \ell - 2(d + 2\ell + 2\log(1/\epsilon)) \geq 10d$. Thus by Lemma 27 we have that

$$(Y_0, Y_0', Y_1, Y_1', Z, Z', X^0, X'^0) \approx_{O(\epsilon)} (U_{3d}, Y_0', U_{d_2}, Y_1', Z, Z', X^0, X'^0),$$

where each U_{3d} is uniform given the previous random variables, but may depend on later random variables. We can now fix (X^0, X'^0, Y^0, Y'^0), and conditioned on this fixing, we have that (X, X') and (Y, Y') are still independent, (X_0, X_0', X_1, X_1') is a deterministic function of (X, X'), and (Y_0, Y_0', Y_1, Y_1') is a deterministic function of (Y, Y'). Further they satisfy the look-ahead properties in the previous two equations. We will ignore the error for now since this only adds at most $O(\epsilon)$ to the final error.

We now claim that conditioned on the fixing of $(X_0, X_0', Y_0, Y_0', Y_1, Y_1')$ (and ignoring the error), the random variables $(V, V', \hat{X}, \hat{X}')$ and $(\hat{Y}, \hat{Y}')$ satisfy the conditions required by Lemma 36. To see this, note that if we fix (Y_0, Y_0', Y_1, Y_1'), then the average conditional

min-entropy of Y is at least $s-\ell-2(d+2\ell+2\log(1/\epsilon))-2\cdot 3d > 2s/3$ as long as c is large enough. Thus by Theorem 22 we have that

$$(\hat{Y}, X_0, X_0') \approx_\epsilon (U_{s/2}, X_0, X_0').$$

Thus conditioned on the further fixing of (X_0, X_0'), we have that $(\hat{Y}, \hat{Y'})$ is a deterministic function of (Y, Y'), and $s/2 \geq d_2$. On the other hand, conditioned on the fixing of (X_0, X_0') and (Y_0, Y_0'), we have X_1 is still close to uniform. Thus by Theorem 41 we have that for any $i \in [L]$,

$$|V_i, Y_1, Y_1' - U_m, Y_1, Y_1'| \leqslant \epsilon$$

and there exists $i \in [L]$ such that

$$|V_i, V_i', Y_1, Y_1' - U_m, V_i', Y_1, Y_1'| \leqslant \epsilon.$$

We now further fix (Y_1, Y_1'). Note that conditioned on this fixing (X, X') and (Y, Y') are still independent. Furthermore (V, V') is now a deterministic function of (X_1, X_1'), and thus independent of (Y, Y'). Finally, note that conditioned on the fixing of (X_0, X_0', X_1, X_1'), the average conditional min-entropy of X is at least $n-\ell-2(d+2\ell+2\log(1/\epsilon))-2\cdot 3m > 2n/3$. Thus by Theorem 22 we have that

$$(\hat{X}, Y_0, Y_0') \approx_\epsilon (U_{n/2}, Y_0, Y_0').$$

Thus conditioned on the further fixing of (Y_0, Y_0'), we have that $(\hat{X}, \hat{X'})$ is a deterministic function of (X, X'), and $n/2 \geq d_1$. Thus, even if conditioned on the fixing of $(X_0, X_0', X_1, X_1', Y_0, Y_0', Y_1, Y_1')$, we have that ($\hat{X}$ is close to $U_{n/2}$. Since (V, V') is obtained from (X_1, X_1', Y_1, Y_1'), we know that ($\hat{X}$ is close to uniform even given $(X_0, X_0', Y_0, Y_0', Y_1, Y_1')$ and (V, V'). Thus by Lemma 36 we have that

$$(\hat{V}, \hat{V'}, Y, Y') \approx (U_m, \hat{V'}, Y, Y'),$$

where the error is $O(L\epsilon)+O(L\epsilon)+O(\epsilon) = O(L\epsilon)$. ◀

Next we give another correlation breaker, which recycles the randomness used.

▶ **Construction 44.** *Inputs:*

- *Let $\ell, m \in \mathcal{N}$ be two integers, $\epsilon > 0$ be an error parameter.*
- *X, Y, two independent sources on n bits with min-entropy at least $n-\ell$.*
- *an advice string $\alpha \in \{0,1\}^L$ and an integer $2 \leq t \leq L$.*
- *An $(L, d_1, d_2, O(L\epsilon))$-NIPM$_y$ that is strong in X.*
- *Let IP be the two source extractor from Theorem 23.*

1. *Let $d' = O(\log(n/\epsilon))$ be the seed length of the extractor from Theorem 22, and let $d = 8\frac{\log L}{\log t}d'$. Let X^0 be a slice of X with length $d+2\ell+2\log(1/\epsilon)$, and Y^0 be a slice of Y with length $d+2\ell+2\log(1/\epsilon)$.*
2. *Compute $Z = \mathsf{IP}(X^0, Y^0)$ and output d bits.*
3. *Use X and Z to do an alternating extraction, and output $3\frac{\log L}{\log t}+1$ random variables $X_0, \cdots, X_{3\frac{\log L}{\log t}}$ where each X_i has d_1 bits.*
4. *Use Y and Z to do an alternating extraction, and output two random variables Y_0, Y_1 where each Y_i has d_2 bits.*
5. *Use X_0, Y_0, α to obtain an $L \times m$ matrix V^0, where for any $i \in [L]$, $V_i^0 = \text{flip-flop}(X_0, Y_0, \alpha_i)$ and outputs m bits.*

6. *For* $i = 1$ *to* $\frac{\log L}{\log t}$ *do the following. Merge every* t *rows of* V^{i-1} *using* NIPM_y *and* (X_{3i-2}, Y_i)*, and output* d' *bits. Concatenate the outputs to become another matrix* W^i*. Note that* W^i *has* L/t^i *rows. Then for every row* $j \in [L/t^i]$*, compute* $V^i_j = \mathsf{Ext}(X_{3i}, W^i_j)$ *to obtain a new matrix* V^i*. Finally let* $Y_{i+1} = \mathsf{Ext}(Y, X_{3i-1})$ *and output* d_2 *bits.*
7. *Output* $\hat{V} = V^{\frac{\log L}{\log t}}$.

We now have the following lemma.

▶ **Lemma 45.** *There exists a costant* $c > 1$ *such that the following holds. Suppose that for any* $t \in \mathcal{N}$ *there exists an* $(t, d_1, d_2, O(t\epsilon))$-$\text{NIPM}_y$ *that is strong in* X *which outputs* $d' = O(\log(n/\epsilon))$ *bits, then there exists an explicit* $(n-\ell, n-\ell, O(L\epsilon))$ *correlation breaker with advice* $\mathsf{AdvCB} : \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^L \to \{0,1\}^m$ *as long as* $d_1 \geq 4m$*,* $m \geq c\log(d_2/\epsilon)$*, and* $n \geq c\frac{\log L}{\log t}\log(n/\epsilon) + max\{8\frac{\log L}{\log t}d_1, 2t \cdot d' + 4d_2\} + 5\ell + 4\log(1/\epsilon)$.

Proof. Throughout the proof we will use letters with prime to denote the corresponding random variables obtained from (X', Y', α'). First, notice that both X^0 and Y^0 have min-entropy at least $d + \ell + 2\log(1/\epsilon)$. Thus by Theorem 23 we have that

$$(Z, X^0) \approx_\epsilon (U_d, X^0)$$

and

$$(Z, Y^0) \approx_\epsilon (U_d, Y^0).$$

We now ignore the error ϵ. Note that conditioned on the fixing of (X^0, X'^0), (Z, Z') is a deterministic function of (Y^0, Y'^0), and thus independent of (X, X'). Moreover, the average conditional min-entropy of X given this fixing is at least $n-\ell-2(d+2\ell+2\log(1/\epsilon)) \geq 8\frac{\log L}{\log t}d_1$ as long as c is large enough. Thus by Lemma 27 (note that the extractor from Z side can use seed length d') we have that

$$\begin{aligned}&(Y^0, Y'^0, Z, Z', X_0, X'_0, \cdots, X_{3\frac{\log L}{\log t}}, X'_{3\frac{\log L}{\log t}})\\ \approx_{O((\frac{\log L}{\log t})^2\epsilon)}\ &(Y^0, Y'^0, Z, Z', U_{d_1}, X'_0, \cdots, U_{d_1}, X'_{3\frac{\log L}{\log t}}),\end{aligned}$$

where each U_{d_1} is uniform given the previous random variables, but may depend on later random variables. Similarly, note that conditioned on the fixing of (Y^0, Y'^0), (Z, Z') is a deterministic function of (X^0, X'^0), and thus independent of (Y, Y'). Moreover, the average conditional min-entropy of Y given this fixing is at least $n - \ell - 2(d + 2\ell + 2\log(1/\epsilon)) \geq 4d_2$. Thus by Lemma 27 we have that

$$(Z, Z', X^0, X'^0, Y_0, Y'_0, Y_1, Y'_1) \approx_{O(\epsilon)} (Z, Z', X^0, X'^0, U_{d_2}, Y'_0, U_{d_2}),$$

where each U_{d_2} is uniform given the previous random variables, but may depend on later random variables. We can now fix (X^0, X'^0, Y^0, Y'^0), and conditioned on this fixing, we have that (X, X') and (Y, Y') are still independent, $(X_0, X'_0, \cdots, X_{3\frac{\log L}{\log t}}, X'_{3\frac{\log L}{\log t}})$ is a deterministic function of (X, X'), and (Y_0, Y'_0, Y_1, Y'_1) is a deterministic function of (Y, Y'). Further they satisfy the look-ahead properties in the previous two equations. We will ignore the error for now since this only adds at most $O((\frac{\log L}{\log t})^2\epsilon)$ to the final error.

Now by Theorem 41 we have that for any $i \in [L]$,

$$|V^0_i, Y_0, Y'_0 - U_m, Y_0, Y'_0| \leqslant \epsilon$$

and there exists $i \in [L]$ such that

$$|V_i^0, V_i'^0, Y_0, Y_0' - U_m, V_i'^0, Y_0, Y_0'| \leqslant \epsilon.$$

We now further fix (Y_0, Y_0'). Note that conditioned on this fixing (X, X') and (Y, Y') are still independent. Furthermore (V^0, V'^0) is now a deterministic function of (X_0, X_0'), and thus independent of (Y, Y'). Thus by the property of NIPM_y we have that for every row j in W^1,

$$(W_j^1, V^0, V'^0, X_1, X_1') \approx_{O(t\epsilon)} (U_{d'}, V^0, V'^0, X_1, X_1'),$$

and there exists a row j such that

$$(W_j^1, W_j'^1, V^0, V'^0, X_1, X_1') \approx_{O(t\epsilon)} (U_{d'}, W_j'^1, V^0, V'^0, X_1, X_1').$$

Note that we have fixed (X^0, X'^0, Y^0, Y'^0), and if we further condition on the fixing of $(X_0, X_0', Y_0, Y_0', X_1, X_1')$, then (W^1, W'^1) is a deterministic function of (Y, Y'). Furthermore (X, X') and (Y, Y') are still independent. We will now use induction to prove the following claim (note that we have already fixed (X^0, X'^0, Y^0, Y'^0)).

▷ Claim 46. Let $T_i = (Y_0, Y_0', X_0, X_0', \cdots, X_{3i-2}, X_{3i-2}')$. In the i' th iteration, the following holds.

1. Conditioned on the further fixing of T_i, we have that (X, X') and (Y, Y') are still independent, and furthermore (W^i, W'^i) is a deterministic function of (Y, Y').
2. For every row j in W^i,

   $$(W_j^i, T_i) \approx_{\epsilon_i} (U_{d'}, T_i),$$

   and there exists a row j such that

   $$(W_j^i, W_j'^i, T_i) \approx_{\epsilon_i} (U_{d'}, W_j'^i, T_i),$$

   where $\epsilon_i = O(\sum_{j=1}^{i} t^j \epsilon)$.

Proof of the claim. The base case of $i = 1$ is already proved above. Now suppose the claim holds for the i'th iteration, we show that it also holds for the $i+1$'th iteration.

To see this, note that conditioned on the fixing of T_i, (X, X') and (Y, Y') are still independent, and furthermore (W^i, W'^i) is a deterministic function of (Y, Y') and thus independent of (X, X'). Note that Y_{i+1} is computed from Y and X_{3i-1} while V^i is computed from X_{3i} and W^i. Thus if we further fix X_{3i-1}, X_{3i-1}' and (W^i, W'^i), then (X, X') and (Y, Y') are still independent, and furthermore Y_{i+1} is a deterministic function of Y and V^i is a deterministic function of X_{3i}. Now W^{i+1} is computed from V^i, X_{3i+1} and Y_{i+1}. Thus if we further fix (X_{3i}, X_{3i}') and (X_{3i+1}, X_{3i+1}') (i.e., we have fixed T_{i+1}) then (X, X') and (Y, Y') are still independent, and furthermore (W^{i+1}, W'^{i+1}) is a deterministic function of (Y, Y').

Next, let h be the row in W^i such that

$$(W_h^i, W_h'^i, T_i) \approx_{\epsilon_i} (U_{d'}, W_h'^i, T_i).$$

Note that V^i has the same number of rows as W^i, and consider the merging of some t rows in V^i that contain row h into W_j^{i+1} (the merging of the other rows is similar and simpler). Without loss of generality assume that these t rows are row $1, 2, \cdots, t$.

First, since for every row j in W^i,

$$(W_j^i, T_i) \approx_{\epsilon_i} (U_{d'}, T_i),$$

and rows h in W^i and W'^i satisfy the independence property, by Theorem 22 (and ignoring the error ϵ_i) we have that for every $j \in [t]$,

$$(V_j^i, T_i, X_{3i-1}, X'_{3i-1}, W_j^i, W_j'^i) \approx_\epsilon (U_m, T_i, X_{3i-1}, X'_{3i-1}, W_j^i, W_j'^i),$$

and

$$(V_h^i, V_h'^i, T_i, X_{3i-1}, X'_{3i-1}, W_j^i, W_j'^i) \approx_\epsilon (U_m, V_h'^i, T_i, X_{3i-1}, X'_{3i-1}, W_j^i, W_j'^i).$$

This is because X_{3i} has average conditional min-entropy at least d_1 even conditioned on the fixing of (X_{3i-1}, X'_{3i-1}). We now ignore the error ϵ. Note that conditioned on the fixing of $(W_j^i, W_j'^i)$, we have that $(V_j^i, V_j'^i)$ is a deterministic function of (X_{3i}, X'_{3i}), and thus independent of (Y, Y'). We now fix $\{(W_j^i, W_j'^i), j \in [t]\}$. Note that conditioned on this fixing $\{V_j^i, j \in [t]\}$ and $\{V_j'^i, j \in [t]\}$ each is a $t \times m$ matrix, and a deterministic function of (X_{3i}, X'_{3i}). Further note that they form two matrices that meet the condition to apply an NIPM. Since $\{(W_j^i, W_j'^i), j \in [t]\}$ is a deterministic function of (Y, Y'), conditioned on this fixing (X, X') and (Y, Y') are still independent. Furthermore the average conditional min-entropy of Y is at least $n - \ell - 2(d + 2\ell + 2\log(1/\epsilon)) - 2d_2 - 2td' \geq 2d_2$. Thus by Theorem 22 we have that

$$(Y_{i+1}, X_{3i-1}) \approx_\epsilon (U_{d_2}, X_{3i-1}).$$

Note that conditioned on the fixing of X_{3i-1}, we have that Y_{i+1} is a deterministic function of Y. Thus we can now further fix (X_{3i-1}, X'_{3i-1}), and conditioned on this fixing, Y_{i+1} is still close to uniform. To conclude, now conditioned on the fixing of $\{(W_j^i, W_j'^i), j \in [t]\}$ and (X_{3i-1}, X'_{3i-1}), we have that $\{V_j^i, j \in [t]\}$ and $\{V_j'^i, j \in [t]\}$ each is a $t \times m$ matrix, and a deterministic function of (X_{3i}, X'_{3i}); Y_{i+1} is still close to uniform and (Y_{i+1}, Y'_{i+1}) is a deterministic function of (Y, Y'). Furthermore X_{3i+1} is close to uniform. Now we can use the property of NIPM_y to show that after merging these t rows, the corresponding row j in W^{i+1} satisfies

$$\begin{aligned}&(W_j^{i+1}, W_j'^{i+1}, T_i, X_{3i-1}, X'_{3i-1}, X_{3i}, X'_{3i}, X_{3i+1}, X'_{3i+1})\\ \approx_{t\epsilon} &(U_{d'}, W_j'^{i+1}, T_i, X_{3i-1}, X'_{3i-1}, X_{3i}, X'_{3i}, X_{3i+1}, X'_{3i+1}).\end{aligned}$$

Adding back all the errors we get that

$$(W_j^{i+1}, W_j'^{i+1}, T_{i+1}) \approx_{\epsilon_{i+1}} (U_{d'}, W_j'^{i+1}, T_{i+1}),$$

where $\epsilon_{i+1} = t\epsilon_i + O(t\epsilon) = O(\sum_{j=1}^{i+1} t^j \epsilon)$. ◁

Now we are basically done. In the last iteration we know that $W^{\frac{\log L}{\log t}}$ has reduced to one row, and $W^{\frac{\log L}{\log t}}$ is close to uniform given $W'^{\frac{\log L}{\log t}}$. Also conditioned on the fixing of $T_{\frac{\log L}{\log t}}$ they are deterministic functions of (Y, Y'). Thus when we use $W^{\frac{\log L}{\log t}}$ to extract $V^{\frac{\log L}{\log t}}$ from $X_{3\frac{\log L}{\log t}}$, by Theorem 22 we have that

$$(\hat{V}, \hat{V}', Y, Y') \approx (U_m, \hat{V}', Y, Y'),$$

where the error is $O(\sum_{j=1}^{\frac{\log L}{\log t}} t^j \epsilon) + O((\frac{\log L}{\log t})^2 \epsilon) = O(L\epsilon)$. ◀

6 The Constructions of Non-Malleable Extractors

In this section we construct our improved seeded non-malleable extractors and seedless non-malleable extractors. Both the constructions follow the general approach developed in recent works [12, 13, 21, 54], i.e., first obtaining an advice and then applying an appropriate correlation breaker with advice. First we need the following advice generator from [12].

▶ **Theorem 47** ([12]). *There exist a constant $c > 0$ such that for all $n > 0$ and any $\epsilon > 0$, there exists an explicit function $\mathsf{AdvGen} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^L$ with $L = c\log(n/\epsilon)$ satisfying the following: Let X be an (n,k)-source, and Y be an independent uniform seed on d bits. Let Y' be a random variable on d bits s.t $Y' \neq Y$, and (Y,Y') is independent of X. Then with probability at least $1-\epsilon$, $\mathsf{AdvGen}(X,Y) \neq \mathsf{AdvGen}(X,Y')$. Moreover, there is a deterministic function g such that $\mathsf{AdvGen}(X,Y)$ is computed as follows. Let Y_1 be a small slice of Y with length $O(\log(n/\epsilon))$, compute $Z = \mathsf{Ext}(X,Y_1)$ where Ext is an optimal seeded extractor from Theorem 22 which outputs $O(\log(n/\epsilon))$ bits. Finally compute $Y_2 = g(Y,Z)$ which outputs $O(\log(1/\epsilon))$ bits and let $\mathsf{AdvGen}(X,Y) = (Y_1,Y_2)$.*

For two independent sources we also have the following slightly different advice generator.

▶ **Theorem 48** ([12]). *There exist constants $0 < \gamma < \beta < 1$ such that for all $n > 0$ and any $\epsilon \geq \epsilon'$ for some $\epsilon' = 2^{-\Omega(n)}$, there exists an explicit function $\mathsf{AdvGen} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^L$ with $L = 2\beta n + O(\log(1/\epsilon))$ satisfying the following: Let X,Y be two independent $(n,(1-\gamma)n)$-sources, and (X',Y') be some tampered versions of (X,Y), such that (X,X') is independent of (Y,Y'). Furthermore either $X \neq X'$ or $Y \neq Y'$. Then with probability at least $1-\epsilon$, $\mathsf{AdvGen}(X,Y) \neq \mathsf{AdvGen}(X',Y')$. Moreover, there is a deterministic function g such that $\mathsf{AdvGen}(X,Y)$ is computed as follows. Let X_1,Y_1 be two small slice of X,Y respectively, with length βn, compute $Z = \mathsf{IP}(X,Y_1)$ where IP is the inner product two source extractor from Theorem 23 which outputs $\Omega(n)$ bits. Finally compute $X_2 = g(X,Z), Y_2 = g(Y,Z)$ which both output $O(\log(1/\epsilon))$ bits and let $\mathsf{AdvGen}(X,Y) = (X_1,X_2,Y_1,Y_2)$.*

By using these advice generators, the general approach of constructing seeded non-malleable extractors and seedless non-malleable extractors can be summarized in the following two theorems.

▶ **Theorem 49.** *[12, 13, 21, 54] There is a constant $c > 1$ such that for any $n,k,d \in \mathcal{N}$ and $\epsilon_1,\epsilon_2 > 0$, if there is a $(k - c\log(n/\epsilon_1), d - c\log(n/\epsilon_1), \epsilon_2)$ advice correlation breaker $\mathsf{AdvCB} : \{0,1\}^k \times \{0,1\}^d \times \{0,1\}^{c\log(n/\epsilon_1)} \to \{0,1\}^m$, then there exists an $(O(k), \epsilon_1+\epsilon_2)$ seeded non-malleable extractor $\mathsf{nmExt} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$. Furthermore if $m \geq c\log(d/\epsilon_1)$ then there exists an $(O(k), \epsilon_1+\epsilon_2)$ seeded non-malleable extractor $\mathsf{nmExt} : \{0,1\}^n \times \{0,1\}^{O(d)} \to \{0,1\}^{\Omega(k)}$.*

Sketch. The seeded non-malleable extractor is constructed as follows. First use the seed and the source to obtain an advice as in Theorem 47 with error $\epsilon_1/3$, however when we compute $Z = \mathsf{Ext}(X,Y_1)$ we in fact output $Z_1 = \mathsf{Ext}(X,Y_1)$ with k bits and choose Z to be a small slice of Z_1 with length $O(\log(n/\epsilon))$. Then we can fix the random variables (Y_1,Y_1',Z,Z',Y_2,Y_2'). Note that conditioned on this fixing (X,X') is still independent of (Y,Y'), and (Z_1,Z_1') is a deterministic function of (X,X') thus is independent of (Y,Y'). Furthermore with probability $1-\epsilon_1/3$, Z_1 has min-entropy at least $k - O(\log(n/\epsilon_1))$ and Y has min-entropy at least $d - O(\log(n/\epsilon_1))$. We can now apply the correlation breaker to (Z_1,Y) and the advice to get the desired output, where the total error is at most $\epsilon_1/3 + \epsilon_1/3 + \epsilon_1/3 + \epsilon_2 = \epsilon_1 + \epsilon_2$. If the output m is large enough (i.e., $m \geq c\log(d/\epsilon_1)$), then we can use it to extract from Y and then extract again from Z_1 to increase the output length to $\Omega(k)$. ◀

▶ **Theorem 50.** *[12, 13, 21, 54] There are constants $c > 1$, $0 < \gamma < \beta < 1/100$ such that for any $n \in \mathcal{N}$ and $\epsilon_1, \epsilon_2 > 0$, if there is a $((1-2\beta)n - c\log(n/\epsilon_1), (1-2\beta)n - c\log(n/\epsilon_1), \epsilon_2)$ advice correlation breaker $\mathsf{AdvCB} : \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^{2\beta n + c\log(1/\epsilon_1)} \to \{0,1\}^m$, then there exists an $((1-\gamma)n, (1-\gamma)n, \epsilon_1 + \epsilon_2)$ non-malleable two source extractor $\mathsf{nmExt} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^m$. Furthermore if $m \geq c\log(n/\epsilon_1)$ then there exists an $((1-\gamma)n, (1-\gamma)n, \epsilon_1 + \epsilon_2)$ non-malleable two source extractor $\mathsf{nmExt} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^{\Omega(n)}$.*

Sketch. The non-malleable two-source extractor is constructed as follows. First use the two independent sources (X, Y) to obtain an advice as in Theorem 48 with error $\epsilon_1/3$, then we can fix the random variables $(X_1, X_1', Y_1, Y_1', X_2, X_2', Y_2, Y_2')$. Note that conditioned on this fixing (X, X') is still independent of (Y, Y'), furthermore with probability $1 - \epsilon_1/3$, both X and Y have min-entropy at least $(1-\gamma)n - \beta n - c\log(1/\epsilon_1) \geq (1-2\beta)n - c\log(1/\epsilon_1)$. We can now apply the correlation breaker to (X, Y) and the advice to get the desired output, where the total error is at most $\epsilon_1/3 + \epsilon_1/3 + \epsilon_1/3 + \epsilon_2 = \epsilon_1 + \epsilon_2$. If the output m is large enough (i.e., $m \geq c\log(d/\epsilon_1)$), then we can use it to extract from Y and then extract again from X to increase the output length to $\Omega(n)$. ◀

Combined with our new correlation breakers with advice, we have the following new constructions of non-malleable extractors.

▶ **Theorem 51.** *There exists a constant $C > 1$ such that for any constant $a \in \mathcal{N}, \dashv \geq \in$, any $n, k \in \mathcal{N}$ and any $0 < \epsilon < 1$ with $k \geq C(\log n + a\log(1/\epsilon))$, there is an explicit construction of a strong seeded (k, ϵ) non-malleable extractor $\{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $d = O(\log n) + \log(1/\epsilon)2^{O(a(\log\log(1/\epsilon))^{\frac{1}{a}})}$ and $m = \Omega(k)$. Alternatively, we can also achieve entropy $k \geq C\log n + \log(1/\epsilon)2^{C\cdot a(\log\log(1/\epsilon))^{\frac{1}{a}}}$ and $d = O(\log n + a\log(1/\epsilon))$.*

Proof. The theorem is obtained by combining Theorem 49, Lemma 43 and Lemma 36. We choose an error ϵ' to be the error in Theorem 49, Lemma 43 and Lemma 36. Thus the total error is $O(L\epsilon')$ where $L = O(\log(n/\epsilon'))$. To ensure $O(L\epsilon') = \epsilon$ it suffices to take $\epsilon' = \frac{\epsilon}{c\log(n/\epsilon)}$ for some constant $c > 1$. We know $\ell = O(\log(n/\epsilon'))$. Therefore to apply Lemma 43 and Lemma 36, we need to find m, d', d_1, d_2 such that

$$d' \geq c(\log\max\{d_1, d_2\} + \log(1/\epsilon')), m \geq d', d_1 \geq 8a\cdot d' + 6m \text{ and } d_2 \geq 8a\cdot d' + c^{a\cdot\log^{\frac{1}{a}} L}\cdot d'.$$

Then we can take

$$k = O(d_1 + m + \ell + \log(1/\epsilon')) \text{ and } d = O(d_2 + m + \ell + \log(1/\epsilon')).$$

It can be seen that we can take $m = O(\log(n/\epsilon'))$, $d' = O(\log\log n + \log(1/\epsilon'))$, $d_1 = 8a\cdot d' + 6m = O(\log n + a\log(1/\epsilon'))$ and $d_2 = 2^{O(a(\log\log(n/\epsilon'))^{\frac{1}{a}})}\cdot d'$. We now consider two cases. First, $\log(1/\epsilon') > \frac{\log n}{c'^{a(\log\log n)^{\frac{1}{a}}}}$ for some large constant c'. In this case we have that

$$\log(1/\epsilon') > \frac{\log n}{c'^{a(\log\log n)^{\frac{1}{a}}}} > \sqrt{\log n}$$

for any $a \geq 2$. Thus

$$\log\log(n/\epsilon')) = \log(\log n + \log(1/\epsilon')) < \log(\log^2(1/\epsilon') + \log(1/\epsilon')) < 2\log\log(1/\epsilon') + 1.$$

Also note that $d' = O(\log\log n + \log(1/\epsilon')) = O(\log(1/\epsilon'))$. Thus in this case we have $d_2 \leq O(\log(1/\epsilon'))2^{O(a(\log\log(1/\epsilon'))^{\frac{1}{a}})} = \log(1/\epsilon')2^{O(a(\log\log(1/\epsilon'))^{\frac{1}{a}})}$. Next, consider the case

where $\log(1/\epsilon') \leq \frac{\log n}{c'a(\log\log n)^{\frac{1}{a}}}$. In this case note that we have $\log(1/\epsilon') < \log n$ and thus $2^{O(a(\log\log(n/\epsilon'))^{\frac{1}{a}})} < 2^{O(a(\log\log(n))^{\frac{1}{a}})}$. Therefore when c' is large enough and $a \geq 2$ we have that

$$d_2 \leq 2^{O(a(\log\log(n))^{\frac{1}{a}})}(\log\log n + \log(1/\epsilon')) \leq \log n.$$

Therefore altogether we have that $d_2 \leq (\log n + \log(1/\epsilon')2^{O(a(\log\log(1/\epsilon'))^{\frac{1}{a}})})$ and $d = O(d_2 + m + \ell + \log(1/\epsilon')) = O(\log n) + \log(1/\epsilon')2^{O(a(\log\log(1/\epsilon'))^{\frac{1}{a}})}$. Note that $\log(1/\epsilon') = \log(1/\epsilon) + \log(\log n + \log(1/\epsilon)) + O(1)$, a careful analysis similar as above shows that we also have that

$$d = O(\log n) + \log(1/\epsilon)2^{O(a(\log\log(1/\epsilon))^{\frac{1}{a}})}.$$

Note that the correlation breaker is completely symmetric to both sources, and the only difference is in generating the advice. Thus after advice generation which costs both sources $O(\log(n/\epsilon))$ entropy, we can switch the role of the seed and the source. Therefore we can also get the other setting of parameters where $k \geq C\log n + \log(1/\epsilon)2^{C\cdot a(\log\log(1/\epsilon))^{\frac{1}{a}}}$ and $d = O(\log n + a\log(1/\epsilon))$. ◀

By using this theorem, we can actually improve the entropy requirement of the non-malleable extractor. Specifically, we have the following theorem.

▶ **Theorem 52.** *There exists a constant $C > 1$ such that for any constant $a \in \mathcal{N}, \dashv \geq \in$, any $n, k \in \mathcal{N}$ and any $0 < \epsilon < 1$ with $k \geq C(\log\log n + a\log(1/\epsilon))$, there is an explicit construction of a strong seeded (k, ϵ) non-malleable extractor $\{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $d = O(\log n) + \log(1/\epsilon)2^{O(a(\log\log(1/\epsilon))^{\frac{1}{a}})}$ and $m = \Omega(k)$. Alternatively, we can also achieve entropy $k \geq C\log\log n + \log(1/\epsilon)2^{C\cdot a(\log\log(1/\epsilon))^{\frac{1}{a}}}$ and $d = O(\log n + a\log(1/\epsilon))$.*

Proof. We start by taking a slice of the seed Y_1 with length $O(\log(n/\epsilon))$ to extract from the source, and output some $k' = 0.9k$ uniform bits with error $\epsilon/2$. Note that conditioned on the fixing of (Y_1, Y_1') where Y_1' is the tampered version, the two sources are still independent, and the seed now has average conditional entropy at least $d - O(\log(n/\epsilon))$. We now switch the role of the seed and the source, and use the output of the extractor from the source as the seed of a non-malleable extractor and apply Theorem 51 with error $\epsilon/2$, so that the final error is ϵ.

Note that now we know the original seed is different from its tampered version, so we only need to obtain advice from the original seed and thus the advice size is $O(\log(d/\epsilon))$. Now we only need

$$k \geq C(\log d + a\log(1/\epsilon))$$

and

$$d - O(\log(n/\epsilon)) \geq C\log k + \log(1/\epsilon)2^{C\cdot a(\log\log(1/\epsilon))^{\frac{1}{a}}}.$$

Thus we can choose

$$k \geq C'(\log\log n + a\log(1/\epsilon))$$

for some slightly larger constant $C' > 1$, while the requirement of the seed is still

$$d = O(\log n) + \log(1/\epsilon)2^{O(a(\log\log(1/\epsilon))^{\frac{1}{a}})}.$$

Similarly, we can switch the role of the seed and the source to get the other setting of parameters. ◀

The next theorem improves the seed length, at the price of using a slightly larger entropy.

▶ **Theorem 53.** *There exists a constant $C > 1$ such that for any $n, k \in \mathcal{N}$ and $0 < \epsilon < 1$ with $k \geq C(\log n + \log(1/\epsilon) \log\log\log(1/\epsilon))$, there is an explicit construction of a strong seeded (k, ϵ) non-malleable extractor $\{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $d = O(\log n + \log(1/\epsilon)(\log\log(1/\epsilon))^2)$ and $m = \Omega(k)$.*

Proof. The theorem is obtained by combining Theorem 49, Lemma 43 and Lemma 38. Again, We choose an error ϵ' to be the error in Theorem 49, Lemma 43 and Lemma 38. Thus the total error is $O(L\epsilon')$ where $L = O(\log(n/\epsilon'))$. To ensure $O(L\epsilon') = \epsilon$ it suffices to take $\epsilon' = \frac{\epsilon}{c \log(n/\epsilon)}$ for some constant $c > 1$. We also know $\ell = O(\log(n/\epsilon'))$ in Lemma 43. Thus to apply Lemma 38, we need to find m, d_1, d_2 such that (for simplicity, we choose $a = 4$ in Lemma 38),

$$m \geq c \log(d_2/\epsilon'), d_1 \geq 20c \log\log L \log(d_2/\epsilon') + 6m \text{ and } d_2 \geq 20c \log^2 L \log(d_1/\epsilon').$$

Then we can take

$$k = O(d_1 + m + \ell + \log(1/\epsilon')) \text{ and } d = O(d_2 + m + \ell + \log(1/\epsilon')).$$

A careful but tedious calculation shows that we can choose $k \geq C(\log n + \log(1/\epsilon') \log\log\log(1/\epsilon'))$ for some large enough constant $C > 1$, and $d = O(\log n + \log(1/\epsilon')(\log\log(1/\epsilon'))^2)$. Note that we can choose $m = O(\log(n/\epsilon'))$ for a large enough constant in $O(.)$, thus by Theorem 49 we can get an output length of $\Omega(k)$. Finally, note that $\log(n/\epsilon') = O(\log(n/\epsilon))$, thus the theorem follows. ◀

Similar to what we have done above, we can also use this to get improved parameters. Specifically, we have

▶ **Theorem 54.** *There exists a constant $C > 1$ such that for any $n, k \in \mathcal{N}$ and $0 < \epsilon < 1$ with $k \geq C(\log\log n + \log(1/\epsilon) \log\log\log(1/\epsilon))$, there is an explicit construction of a strong seeded (k, ϵ) non-malleable extractor $\{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $d = O(\log n + \log(1/\epsilon)(\log\log(1/\epsilon))^2)$ and $m = \Omega(k)$. Alternatively, we can also achieve entropy $k \geq C(\log\log n + \log(1/\epsilon)(\log\log(1/\epsilon))^2)$ and seed length $d = O(\log n + \log(1/\epsilon) \log\log\log(1/\epsilon))$.*

For non-malleable two-source extractors we have the following theorem.

▶ **Theorem 55.** *There exists a constant $0 < \gamma < 1$ and a non-malleable two-source extractor for $(n, (1-\gamma)n)$ sources with error $2^{-\Omega(n \log\log n / \log n)}$ and output length $\Omega(n)$.*

Proof. The theorem is obtained by combining Theorem 50, Lemma 45 and Lemma 36. Again, we choose an error ϵ' to be the error in Theorem 49, Lemma 43 and Lemma 38. Thus the total error is $O(L\epsilon')$ where $L = O(n)$. To ensure $O(L\epsilon') = \epsilon$ it suffices to take $\epsilon' = \frac{\epsilon}{cn}$ for some constant c. We also know $\ell = 2\beta n + o(n)$ for some constant $\beta < 1/100$ in Lemma 45. We choose $a = 2$ in Lemma 36 and thus we obtain a correlation breaker with $m = O(\log(n/\epsilon'))$, $d_1 = O(\log(n/\epsilon'))$ and $d_2 = \log(n/\epsilon')2^{O(\sqrt{\log t})}$ where t is the parameter in Construction 44 with $t \leq L$. Note that this also satisfies that $d_1 \geq 4m$ and $m \geq c \log(d_2/\epsilon)$ as required by Lemma 45.

Now we need to ensure that

$$(1-\beta)n \geq c\frac{\log L}{\log t} \log(n/\epsilon') + max\{8\frac{\log L}{\log t} d_1, 2t \cdot d' + 4d_2\} + 5\ell + 4\log(1/\epsilon'),$$

where $d' = O(\log(n/\epsilon'))$. We choose $t = \frac{\log L}{\log\log L}$ and this gives us

$$(1-12\beta)n \geq C\frac{\log L}{\log\log L}\log(n/\epsilon'),$$

for some constant $C > 1$. Note that $\log(n/\epsilon') = O(\log(n/\epsilon))$ thus we can set $\epsilon = 2^{-\Omega(n\log\log n/\log n)}$ and satisfy the above inequality. ◀

For applications in two-source extractors, we first need the following generalization of non-malleable extractors, which allows multiple tampering.

▶ **Definition 56** (Seeded t-Non-malleable extractor). *A function* $\mathrm{snmExt} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *is a seeded t-non-malleable extractor for min-entropy k and error ϵ if the following holds : If X is a source on $\{0,1\}^n$ with min-entropy k and $\mathcal{A}_1, \cdots, \mathcal{A}_t : \{0,1\}^d \to \{0,1\}^d$ are t arbitrary tampering functions with no fixed points, then*

$$\big|\mathrm{snmExt}(X, U_d) \circ \{\mathrm{snmExt}(X, \mathcal{A}_i(U_d)), i \in [t]\} \circ U_d - U_m \circ \{\mathrm{snmExt}(X, \mathcal{A}_i(U_d)), i \in [t]\} \circ U_d\big| < \epsilon$$

where U_m is independent of U_d and X.

The following theorem is a special case of Theorem 8.6 proved in [54].

▶ **Theorem 57.** *Suppose there is a function f, a constant $\gamma > 0$ and an explicit non-malleable two-source extractor for $(f(\epsilon), (1-\gamma)f(\epsilon))$ sources with error ϵ and output length $\Omega(f(\epsilon))$. Then there is a constant $C > 0$ such that for any $0 < \epsilon < 1$ with $k \geq Ct^2(\log n + f(\epsilon))$, there is an explicit strong seeded t-non-malleable extractor for (n,k) sources with seed length $d = Ct^2(\log n + f(\epsilon))$, error $O(t\epsilon)$ and output length $\Omega(f(\epsilon))$.*

Combined with Theorem 55, this immediately gives the following theorem.

▶ **Theorem 58.** *There is a constant $C > 0$ such that for any $0 < \epsilon < 1$ and $n, k \in \mathcal{N}$ with $k \geq Ct^2(\log n + \frac{\log(1/\epsilon)\log\log(1/\epsilon)}{\log\log\log(1/\epsilon)})$, there is an explicit strong seeded t-non-malleable extractor for (n,k) sources with seed length $d = Ct^2(\log n + \frac{\log(1/\epsilon)\log\log(1/\epsilon)}{\log\log\log(1/\epsilon)})$, error $O(t\epsilon)$ and output length $\Omega(k/t^2)$. As a special case, there exists a seeded non-malleable extractor for entropy $k \geq C(\log n + \frac{\log(1/\epsilon)\log\log(1/\epsilon)}{\log\log\log(1/\epsilon)})$ and seed length $d = C(\log n + \frac{\log(1/\epsilon)\log\log(1/\epsilon)}{\log\log\log(1/\epsilon)})$.*

Similar techniques as above can reduce the $\log n$ term in the entropy requirement to $\log\log n$, so we get

▶ **Theorem 59.** *There is a constant $C > 0$ such that for any $0 < \epsilon < 1$ and $n, k \in \mathcal{N}$ with $k \geq C(\log\log n + \frac{\log(1/\epsilon)\log\log(1/\epsilon)}{\log\log\log(1/\epsilon)})$, there is an explicit strong seeded non-malleable extractor for (n,k) sources with seed length and seed length $d = C(\log n + \frac{\log(1/\epsilon)\log\log(1/\epsilon)}{\log\log\log(1/\epsilon)})$.*

Ben-Aroya et. al [8] proved the following theorem.

▶ **Theorem 60** ([8]). *Suppose there is a function f and an explicit strong seeded t-non-malleable extractor (n,k') sources with seed length and entropy requirement $d = k' = f(t,\epsilon)$, then for every constant $\epsilon > 0$ there exist constants $t = t(\epsilon), c = c(\epsilon)$ and an explicit extractor $\mathsf{Ext} : (\{0,1\}^n)^2 \to \{0,1\}$ for two independent (n,k) sources with $k \geq f(t, 1/n^c)$ and error ϵ.*

Combined with Theorem 53, this immediately gives the following theorem.

▶ **Theorem 61.** *For every constant $\epsilon > 0$, there exists a constant $C > 1$ and an explicit two source extractor $\mathsf{Ext} : (\{0,1\}^n)^2 \to \{0,1\}$ for entropy $k \geq C\frac{\log n\log\log n}{\log\log\log n}$ with error ϵ.*

7 Non-Malleable Two-Source Extractor and Non-Malleable Code

Formally, non-malleable codes are defined as follows.

▶ **Definition 62** ([1]). *Let* NM_k *denote the set of trivial manipulation functions on* k*-bit strings, which consists of the identity function* $I(x) = x$ *and all constant functions* $f_c(x) = c$*, where* $c \in \{0,1\}^k$*. Let* $E : \{0,1\}^k \to \{0,1\}^m$ *be an efficient randomized* encoding *function, and* $D : \{0,1\}^m \to \{0,1\}^k$ *be an efficient deterministic* decoding *function. Let* $\mathcal{F} : \{0,1\}^m \to \{0,1\}^m$ *be some class of functions. We say that the pair* (E, D) *defines an* $(\mathcal{F}, k, \epsilon)$-non-malleable code*, if for all* $f \in \mathcal{F}$ *there exists a probability distribution* G *over* NM_k*, such that for all* $x \in \{0,1\}^k$*, we have*

$$|D(f(E(x))) - G(x)| \leq \epsilon.$$

▶ Remark 63. The above definition is slightly different form the original definition in [36]. However, [1] shows that the two definitions are equivalent.

We will mainly be focusing on the following family of tampering functions in this paper.

▶ **Definition 64.** *Given any* $t > 1$*, let* $\mathcal{S}_n^t$ *denote the tampering family in the* t-split-state-model*, where the adversary applies* t *arbitrarily correlated functions* $h_1, \cdots, h_t$ *to* t *separate,* n*-bit parts of string. Each* h_i *can only be applied to the* i*-th part individually.*

We remark that even though the functions $h_1, \cdots, h_t$ can be correlated, their correlation is independent of the original codewords. Thus, they are actually a convex combination of independent functions, applied to each part of the codeword. Therefore, without loss of generality we can assume that each h_i is a deterministic function, which acts on the i-th part of the codeword individually.We will mainly consider the case of $t = 2$, i.e., the two-split-state model. We recall the original definition of non-malleable two-source extractors by Cheraghchi and Gursuswami [18]. First we define the following function.

$$\mathrm{copy}(x, y) = \begin{cases} x & \text{if } x \neq same^\star \\ y & \text{if } x = same^\star \end{cases}$$

▶ **Definition 65** (Seedless Non-Malleable 2-Source Extractor). *A function* $\mathsf{nmExt} : (\{0,1\}^n)^2 \to \{0,1\}^m$ *is a* (k, ϵ)*-seedless non-malleable extractor for two independent sources, if it satisfies the following property: Let* X, Y *be two independent* (n, k) *sources, and* $f_1, f_2 : \{0,1\}^n \to \{0,1\}^n$ *be two arbitrary tampering functions, then*

1. $|\mathsf{nmExt}(X, Y) - U_m| \leq \epsilon$.
2. *There is a distribution* $\mathcal{D}$ *over* $\{0,1\}^m \cup \{same^\star\}$ *such that for an independent* Z *sampled from* D*, we have*

$$(\mathsf{nmExt}(X, Y), \mathsf{nmExt}(f_1(X), f_2(Y))) \approx_\epsilon (\mathsf{nmExt}(X, Y), \mathrm{copy}(Z, \mathsf{nmExt}(X, Y))).$$

Cheraghchi and Gursuswami [18] showed that the relaxed definition 5 implies the above general definition with a small loss in parameters. Specifically, we have

▶ **Lemma 66** ([18]). *Let* nmExt *be a* $(k - \log(1/\epsilon), \epsilon)$*-non-malleable two-source extractor according to Definition 5. Then* nmExt *is a* $(k, 4\epsilon)$*-non-malleable two-source extractor according to Definition 65.*

The following theorem was proved by Cheraghchi and Gursuswami [18], which establishes a connection between seedless non-malleable extractors and non-malleable codes.

▶ **Theorem 67.** *Let* $\mathsf{nmExt} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^m$ *be a polynomial time computable seedless* 2*-non-malleable extractor at min-entropy* n *with error* ϵ*. Then there exists an explicit non-malleable code with an efficient decoder in the* 2*-split-state model with block length* $= 2n$*, rate* $= \frac{m}{2n}$ *and error* $= 2^{m+1}\epsilon$*.*

One can construct a non-malleable code in the 2-split-state model from a non-malleable two-source extractor as follows: Given any message $s \in \{0,1\}^m$, the encoding $\mathsf{Enc}(s)$ is done by outputting a uniformly random string from the set $\mathsf{nmExt}^{-1}(s) \subset \{0,1\}^{2n}$. Given any codeword $c \in \{0,1\}^{2n}$, the decoding $\mathsf{Dec}(c)$ is done by outputting $\mathsf{nmExt}(c)$. Thus, to get an efficient encoder we need a way to efficiently uniformly sample from the pre-image of any output of the extractor.

Since our new non-malleable two-source extractor follows the same structure as in [54], we can use the same sampling procedure there to efficiently uniformly sample from the pre-image of any output of the extractor. We briefly recall the construction and sampling procedure in [54].

The extractor construction and sampling

The high level structure of the non-malleable two-source extractor in [54] is as follows. First take two small slices (X_1, Y_1) of both sources and apply the inner product based two-source extractor, as in Theorem 23. Then, use the output to sample $O(\log(1/\epsilon))$ bits from the encodings of both sources, using a randomness efficient sampler and an asymptotically good linear encoding of the sources. We need an asymptotically good encoding since then we only need to sample $O(\log(1/\epsilon))$ bits to ensure that the sampling of two different codewords are different with probability at least $1 - \epsilon$. The advice is then obtained by combining the slices and the sample bits. Now, take two larger slices (X_2, Y_2) of both sources and apply the correlation breaker. Finally, take another larger slice of either source (say X_3 from X) and apply a strong linear seeded extractor, which is easy to invert and has the same pre-image size for any output. By limiting the size of each slice to be small, the construction ensures that there are at least $n/2$ bits of each source that are only used in the encoding of the sources but never used in the subsequent extraction.

Now to sample uniformly from the pre-image of any output, we first uniformly independently generate the slices (X_1, Y_1, X_2, Y_2) and the sampled bits Z. From these we can compute the coordinates of the sampled bits and the output of the correlation breaker. Now we can invert the linear seeded extractor and uniformly sample X_3 given the output of the extractor and the output of the correlation breaker (which is used as the seed of the linear seeded extractor). Now, to sample the rest of the bits, we need to condition on the event that the sample bits from the encoding of the sources are indeed Z. Note that Z has size at most αn for some small constant $\alpha < 1/2$ since we can restrict the error to be at least some $2^{-\Omega(n)}$. Also note that for each source we have already sampled some bits but there are still at least $n/2$ un-sampled free bits, thus we insist on that no matter which αn columns of the generating matrix of the encoding we look at, the sub matrix corresponding to these columns and the last $n/2$ rows have full column rank. If this is true then no matter which coordinates we use and what Z is, the pre-image always have the same size and we can uniformly sample from the pre-image by solving a system of linear equations.

In [54], we use the Reed-Solomn encoding for each source with field $\mathbf{F}_q$ for $q \approx n$. This is asymptotically good and also satisfies the property that any sub matrix with less columns than rows has full column rank since it is a Vandermonde matrix. However in this case each symbol has roughly $\log n$ bits so we can sample at most $n/\log n$ symbols (otherwise fixing them may already cost us all the entropy), thus the best error we can get using this encoding is $2^{-n/\log n}$.

We now give a new construction of non-malleable two-source extractors for two $(n, (1-\gamma)n)$ sources, where $0 < \gamma < 1$ is some constant. First, we need the following ingredients.

▶ **Theorem 68** ([54]). *There exists a constant $0 < \alpha < 1$ such that for any $n \in \mathcal{N}$ and $2^{-\alpha n} < \epsilon < 1$ there exists a linear seeded strong extractor $\mathsf{IExt} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{0.3d}$ with $d = O(\log(n/\epsilon))$ and the following property. If X is a $(n, 0.9n)$ source and R is an independent uniform seed on $\{0,1\}^d$, then*

$$|(\mathsf{IExt}(X,R), R) - (U_{0.3d}, R)| \leq \epsilon.$$

Furthermore for any $s \in \{0,1\}^{0.3d}$ and any $r \in \{0,1\}^d$, $|\mathsf{IExt}(\cdot, r)^{-1}(s)| = 2^{n-0.3d}$.

▶ **Definition 69** (Averaging sampler [64]). *A function $\mathsf{Samp} : \{0,1\}^r \to [n]^t$ is a (μ, θ, γ) averaging sampler if for every function $f : [n] \to [0,1]$ with average value $\frac{1}{n}\sum_i f(i) \geqslant \mu$, it holds that*

$$\Pr_{i_1,\ldots,i_t \leftarrow \mathsf{Samp}(U_R)} \left[\frac{1}{t} \sum_i f(i) < \mu - \theta \right] \leq \gamma.$$

Samp has distinct samples if for every $x \in \{0,1\}^r$, the samples produced by $\mathsf{Samp}(x)$ are all distinct.

▶ **Theorem 70** ([64]). *Let $1 \geq \delta \geq 3\tau > 0$. Suppose that $\mathsf{Samp} : \{0,1\}^r \to [n]^t$ is an (μ, θ, γ) averaging sampler with distinct samples for $\mu = (\delta - 2\tau)/\log(1/\tau)$ and $\theta = \tau/\log(1/\tau)$. Then for every δn-source X on $\{0,1\}^n$, the random variable $(U_r, X_{Samp(U_r)})$ is $(\gamma + 2^{-\Omega(\tau n)})$-close to (U_r, W) where for every $a \in \{0,1\}^r$, the random variable $W|_{U_r=a}$ is $(\delta - 3\tau)t$-source.*

▶ **Theorem 71** ([64]). *For every $0 < \theta < \mu < 1$, $\gamma > 0$, and $n \in \mathcal{N}$, there is an explicit (μ, θ, γ) averaging sampler $\mathsf{Samp} : \{0,1\}^r \to [n]^t$ that uses*

- *t distinct samples for any $t \in [t_0, n]$, where $t_0 = O(\frac{1}{\theta^2}\log(1/\gamma))$, and*
- *$r = \log(n/t) + \log(1/\gamma)\mathsf{poly}(1/\theta)$ random bits.*

7.1 A new advice generator

Here we show that we can give a new advice generator with optimal advice length. We have the following construction. Let (X, Y) be two independent $(n, (1-\tau)n)$ sources. Let IP be the inner product two-source extractor from Theorem 23, and Samp : be the sampler from Theorem 70. Let $L > 0$ be a parameter, and $c > 0$ be a constant to be chosen later. We have the following algorithm.

1. Let $n_1 = 3\tau n$. Divide X into $X = (X_1, X_2)$ such that X_1 has n_1 bits and X_2 has $n_2 = (1-3\tau)n$ bits. Similarly divide Y into $Y = (Y_1, Y_2)$ such that Y_1 has n_1 bits and Y_2 has $n_2 = (1-3\tau)n$ bits.
2. Compute $Z = \mathsf{IP}(X_1, Y_1)$ which outputs $r = \Omega(n) \leq \tau n$ bits.
   item Let $\mathbf{F}$ be the finite field $\mathbf{F}_{2^{\log n}}$. Let $n_0 = \frac{n_2}{\log n}$. Let $\mathsf{RS} : \mathbf{F}^{n_0} \to \mathbf{F}^n$ be the Reed-Solomon code encoding n_0 symbols of $\mathbf{F}$ to n symbols in $\mathbf{F}$ (we slightly abuse the use of RS to denote both the code and the encoder). Thus RS is a $[n, n_0, n - n_0 + 1]_n$ error correcting code. Let $\hat{X}_2$ be X_2 written backwards, and similarly $\hat{Y}_2$ be Y_2 written backwards. Let $\overline{X}_2 = \mathsf{RS}(\hat{X}_2)$ and $\overline{Y}_2 = \mathsf{RS}(\hat{Y}_2)$.
3. Use Z to sample $r/\log n$ distinct symbols from $\overline{X}_2$ (i.e., use each $\log n$ bits to sample a symbol), and write the symbols as a binary string $\tilde{X}_2$. Note that $\tilde{X}_2$ has r bits. Similarly, use Z to sample $r/\log n$ distinct symbols from $\overline{Y}_2$ and obtain a binary string $\tilde{Y}_2$ with r bits.

4. Let $V_1 = X_1 \circ Y_1 \circ \tilde{X}_2 \circ \tilde{Y}_2$.
5. Take a slice of X_2 with length $15\tau n$, and let it be X_3. Similarly, take a slice of Y_2 with length $10\tau n$, and let it be Y_3. Compute $W = \mathsf{IP}(X_3, Y_3)$ which outputs $r = \Omega(n) \leq \tau n$ bits.
6. Take a slice of X_2 with length $40\tau n$, and let it be X_4. Use W and X_4 to do an alternating extraction protocol for $L = \log^* n$[6] rounds, and output $(R_1, \cdots, R_L) = \mathsf{IaExt}(X_4, W)$, where each S_i, R_i used in the alternating extraction has $\tau n / \log n$ bits.
7. Set $i = 1$ and let n_1 be the length of V_1, which is at most $8\tau n$. While $L < c \log n_i$ do the following: encode V_i to $\tilde{V}_i$ using an asymptotically good binary error correcting code. Cut R_i into $O(\log n_i)$ bits. Use the sampler from Theorem 71 and R_i to sample $\log n_i$ bits of $\tilde{V}_i$, let the sampled string be $\overline{V_i}$. Set $V_{i+1} = R_i \circ \overline{V_i}$ and let $i = i + 1$.
8. Finally, cut R_i into $O(\log n_i)$ bits. Use the sampler from Theorem 71 and R_i to sample $L - |R_i|$ bits of $\tilde{V}_i$, let the sampled string be $\overline{V_i}$. Set $\tilde{\alpha} = R_i \circ \overline{V_i}$ which has length L.

We have the following lemma.

▶ **Lemma 72.** *There are constants $0 < \tau, \mu < 1$ and $C > 1$ such that the following holds. Let (X, Y) be two independent $(n, (1-\tau)n)$ sources, and (X', Y') be their tampered versions. Assume that either the tampering function f on X or the tampering function g on Y has no fixed point. For any L such that $C \leq L \leq \frac{\mu n}{\log n}$, with probability $1 - 2^{-\Omega(L)}$ over the fixing of $(X_1, Y_1, \tilde{X}_2, \tilde{Y}_2, X_3, Y_3, X_4)$ and the tampered versions $(X_1', Y_1', \tilde{X}_2', \tilde{Y}_2', X_3', Y_3', X_4')$, we have that $\tilde{\alpha} \neq \tilde{\alpha}'$. Moreover, conditioned on these fixings, X and Y are independent, and the average conditional min-entropy of both X and Y is $(1 - O(\tau))n$.*

Proof. As usual we use letters with primes to denote the tampered versions of random variables. First note that both X_1 and Y_1 have min-entropy at least $2\tau n$, thus by Theorem 23, we have that

$$(Z, X_1) \approx_{2^{-\Omega(n)}} (U, X_1),$$

and

$$(Z, Y_1) \approx_{2^{-\Omega(n)}} (U, Y_1).$$

If $X_1 \neq X_1'$ or $Y_1 \neq Y_1'$ then we have $V_1 \neq V_1'$. Now consider the case where $X_1 \neq X_1'$ and $Y_1 \neq Y_1'$. In this case we have $Z = Z'$ and either $X_2 \neq X_2'$ or $Y_2 \neq Y_2'$. Without loss of generality assume that $X_2 \neq X_2'$. We can now first fix (X_1, X_1'). Note that conditioned on this fixing, $Z = Z'$ is a deterministic function of Y, and thus independent of (X_2, X_2'). The Reed-Solomon encoding of $\hat{X}_2$ and $\hat{X}_2'$ ensures that $\overline{X}_2$ and $\overline{X}_2'$ differ in at least $n - n_0 + 1 > 0.9n$ symbols. Thus, with probability $1 - 2^{-\Omega(n)} - 2^{-\Omega(r/\log n)} = 1 - 2^{-\Omega(n/\log n)}$ over Z, we have that $\tilde{X}_2 \neq \tilde{X}_2'$. Therefore, altogether with probability $1 - 2^{-\Omega(n/\log n)}$ over the fixing of $(X_1, Y_1, \tilde{X}_2, \tilde{Y}_2)$ and $(X_1', Y_1', \tilde{X}_2', \tilde{Y}_2')$ we have that $V_1 \neq V_1'$.

We now fix $(X_1, Y_1, \tilde{X}_2, \tilde{Y}_2)$ and $(X_1', Y_1', \tilde{X}_2', \tilde{Y}_2')$. Note that conditioned on this fixing, X and Y are independent. Moreover, the average conditional min-entropy of both X_3 and Y_3 is at least $15\tau n - \tau n - 2\tau n - 3\tau n = 9\tau n$. Thus by Theorem 23, we have that

$$(W, X_3) \approx_{2^{-\Omega(n)}} (U, X_3).$$

[6] Here by $\log^* n$ we mean the number of steps it takes to get down to c' by computing $n \to c \log n$ for some constants c, c'.

We ignore the error for now since this only adds $2^{-\Omega(n)}$ to the final error. We now fix (X_3, X_3'). Note that conditioned on this fixing, (W, W') is a deterministic function of (Y, Y'), and thus independent of (X, X'). Further, the average conditional min-entropy of X_4 is at least $40\tau n - \tau n - 2(15\tau n + \tau n) - 3\tau n = 4\tau n$. Thus by Lemma 27 we have that for any $0 \leq j \leq L-1$,

$$(W, W', \{R_1, R_1', \cdots, R_j, R_j'\}, R_{j+1}) \approx_{\epsilon'} (W, W', \{R_1, R_1', \cdots, R_j, R_j'\}, U),$$

where $\epsilon' = O(L2^{-\Omega(n/\log n)}) = 2^{-\Omega(n/\log n)}$. Since conditioned on the fixing of (W, W'), the random variables $\{R_i, R_i'\}$ are deterministic functions of (X, X') and independent of (Y, Y'), we also have that

$$(Y_3, Y_3', \{R_1, R_1', \cdots, R_j, R_j'\}, R_{j+1}) \approx_{\epsilon'} (Y_3, Y_3', \{R_1, R_1', \cdots, R_j, R_j'\}, U).$$

We now further fix (Y_3, Y_3'). Note that now we have fixed $(X_1, Y_1, \tilde{X}_2, \tilde{Y}_2, X_3, Y_3)$ and $(X_1', Y_1', \tilde{X}_2', \tilde{Y}_2', X_3', Y_3')$. Ignoring the error for now let's assume that $V_1 \neq V_1'$ (note that (V_1, V_1') are now fixed) and for any $0 \leq j \leq L-1$,

$$(\{R_1, R_1', \cdots, R_j, R_j'\}, R_{j+1}) = (\{R_1, R_1', \cdots, R_j, R_j'\}, U).$$

Let j be the index when the protocol executes step 8. We know that $j \leq L$ since in each step the length of the string V_i goes from n_i to $O(\log n_i)$. We have the following observation. For any $1 \leq i \leq j$, we have that V_i is a deterministic function of $(R_1, \cdots, R_{i-1})$; similarly, V_i' is a deterministic function of $(R_1', \cdots, R_{i-1}')$. Next, we have the following claim.

▷ Claim 73. For any $1 \leq i < j$, suppose that conditioned on the fixing of $(R_1, \cdots, R_{i-1}), (R_1', \cdots, R_{i-1}')$ we have $V_i \neq V_i'$, then with probability $1 - 2^{-\Omega(\log n_i)}$ over the further fixing of (R_i, R_i'), we have $V_{i+1} \neq V_{i+1}'$. Suppose that conditioned on the fixing of $(R_1, \cdots, R_{j-1}), (R_1', \cdots, R_{j-1}')$ we have $V_j \neq V_j'$, then with probability $1 - 2^{-\Omega(L)}$ over the further fixing of (R_j, R_j'), we have $\tilde{\alpha} \neq \tilde{\alpha}'$.

Proof of the claim. Suppose that conditioned on the fixing of $(R_1, \cdots, R_{i-1}), (R_1', \cdots, R_{i-1}')$ we have $V_i \neq V_i'$. Note that now (V_i, V_i') is also fixed. We know that R_i is still uniform. Again, we have two cases. First, if $R_i \neq R_i'$, then we definitely have $V_{i+1} \neq V_{i+1}'$. Otherwise, we have $R_i = R_i'$. The encoding of V_i and V_i' ensures that at least a constant fraction of bits in $\tilde{V}_i$ and $\tilde{V}_i'$ are different. Thus by Theorem 71 with probability $1 - 2^{-\Omega(\log n_i)}$ over the further fixing of (R_i, R_i'), we have that $\overline{V_i} \neq \overline{V_i}'$ and thus $V_{i+1} \neq V_{i+1}'$.

For the case of $i = j$, the argument is the same, except now we are sampling $L - O(\log n_j)$ bits, and the probability that $\overline{V_i} \neq \overline{V_i}'$ is $2^{-\Omega(L - O(\log n_j))} = 2^{-\Omega(L)}$ since $L \geq c \log n_j$. ◁

Now we are basically done. Since we start with $V_1 \neq V_1'$, at the end the probability that $\tilde{\alpha} \neq \tilde{\alpha}'$ is at least

$$\Pi_{i=1}^{j-1}(1 - 2^{-\Omega(\log n_i)}) \cdot (1 - 2^{-\Omega(L)}).$$

Note that for any $1 \leq i < j$ we have $n_{i+1} = O(\log n_i)$, so $2^{-\Omega(\log n_i)} \leq 2^{-\Omega(\log n_i)}/2$. Thus the terms $2^{-\Omega(\log n_i)}$ form at least a geometric expression and hence this probability is at least $1 - O(2^{-\Omega(L)}) = 1 - 2^{-\Omega(L)}$. Adding back all the errors, and noticing that $C \leq L \leq \frac{\mu n}{\log n}$ for some properly chosen constants C and μ, the final error is still $1 - 2^{-\Omega(L)}$. Moreover, since the size of each random variable in $(X_1, Y_1, \tilde{X}_2, \tilde{Y}_2, X_3, Y_3, X_4)$ is at most $O(\tau n)$, conditioned on the fixing of $(X_1, Y_1, \tilde{X}_2, \tilde{Y}_2, X_3, Y_3, X_4)$ and the tampered versions $(X_1', Y_1', \tilde{X}_2', \tilde{Y}_2', X_3', Y_3', X_4')$, the average conditional min-entropy of both X and Y is $(1 - O(\tau))n$. ◀

We now use the above advice generator to give a new construction of non-malleable two-source extractors. Let (X, Y) be two independent $(n, (1-\gamma)n)$ sources with $\gamma \leq \tau$ where τ is the constant in Lemma 72.

- Let AdvGen be the advice generator from Lemma 72 for some error ϵ_1.
- Let AdvCB be the correlation breaker with advice from Lemma 45 with error some ϵ_2, using the merger from Lemma 36.
- Let IExt be the invertible linear seeded extractor form Theorem 68.

1. Compute $\tilde{\alpha} = \mathsf{AdvGen}(X, Y)$.
2. Consider the unused part of X. Divide it into (X_5, X_6, X_7) where X_5, X_6 has length $\alpha n, \beta n$ for some constants $\beta > \alpha > \gamma$, and X_7 is the rest of X with length at least $n/2$. Similarly, divide the unused part of Y into (Y_5, Y_6, Y_7) where Y_5, Y_6 has length $\alpha n, \beta n$ and Y_7 is the rest of Y with length at least $n/2$ (this can be ensured by choosing α, β, γ to be small enough).
3. Compute $V = \mathsf{AdvCB}(X_5, Y_5, \tilde{\alpha})$ which outputs $d = O(\log(n/\epsilon_2))$ bits.
4. Finally compute $W = \mathsf{IExt}(Y_6, V)$ which outputs $\Omega(n)$ bits.

We need the following proposition.

▶ **Proposition 74** ([18]). *Let D and D' be distributions over the same finite space Ω, and suppose they are ϵ-close to each other. Let $E \subseteq \Omega$ be any event such that $D(E) = p$. Then, the conditional distributions $D|E$ and $D'|E$ are (ϵ/p)-close.*

We now have the following theorem.

▶ **Theorem 75.** *Assume that either the tampering function f on X or the tampering function g on Y has no fixed point. There exist a constant $C > 1$ such that as long as $n \geq C\frac{\log\log(1/\epsilon_1)}{\log\log\log(1/\epsilon_1)}\log(n/\epsilon_2)$, the above non-malleable two-source extractor gives a non-malleable code with error $\epsilon_1 + O(\log(1/\epsilon_1)\sqrt{\epsilon_2})$ and rate $\Omega(\log(1/\epsilon_2)/n)$.*

Proof. First note that by Lemma 72, conditioned on the fixing of $H = (X_1, Y_1, \tilde{X}_2, \tilde{Y}_2, X_3, Y_3, X_4)$ and the tampered versions $H' = (X'_1, Y'_1, \tilde{X}'_2, \tilde{Y}'_2, X'_3, Y'_3, X'_4)$, X and Y are independent, and the average conditional min-entropy of both X and Y is $(1 - O(\gamma))n$. If in addition we have that $\tilde{\alpha} \neq \tilde{\alpha}'$, then we will apply Lemma 45 and Lemma 36. Note that in order to set the error of the advice generator to be ϵ_1, we need to set the advice length to be $L = O(\log(1/\epsilon_1))$ by Lemma 72. Thus in Lemma 45 we need to merge $L = O(\log(1/\epsilon_1))$ rows.

Again, as in Theorem 55, we know that when we apply the correlation breaker to X_5 and Y_5, the entropy loss of both of them is $O(\gamma n)$. By choosing α, β, γ appropriately we can ensure that X_5 and Y_5 have sufficient entropy in them. We choose $a = 2$ in Lemma 36 and thus we obtain a correlation breaker with $m = O(\log(n/\epsilon_2))$, $d_1 = O(\log(n/\epsilon_2))$ and $d_2 = \log(n/\epsilon_2)2^{O(\sqrt{\log t})}$ where t is the parameter in Construction 44 with $t \leq L$. Note that this also satisfies that $d_1 \geq 4m$ and $m \geq c\log(d_2/\epsilon)$ as required by Lemma 45.

Now we need to ensure that

$$(\alpha - O(\gamma))n \geq c\frac{\log L}{\log t}\log(n/\epsilon_2) + max\{8\frac{\log L}{\log t}d_1, 2t \cdot d' + 4d_2\} + 5\ell + 4\log(1/\epsilon_2),$$

where $d' = O(\log(n/\epsilon_2))$. We choose $t = \frac{\log L}{\log\log L}$ and this gives us

$$n \geq C\frac{\log L}{\log\log L}\log(n/\epsilon_2),$$

for some constant $C > 1$. That is, we need

$$n \geq C \frac{\log\log(1/\epsilon_1)}{\log\log\log(1/\epsilon_1)} \log(n/\epsilon_2),$$

for some constants $C > 1$. As long as this condition is satisfied, conditioned on the event that $\tilde{\alpha} \neq \tilde{\alpha}'$, we have that

$$(V, V', H, H', X, X') \approx_{O(L\epsilon_2)} (U, V', H, H', X, X').$$

By choosing $\beta > \alpha$ appropriately, we can ensure that conditioned on the fixing of the previous random variables in the computation, Y_6 has entropy $\Omega(n)$ and (V, V') is a deterministic function of (X, X') and thus independent of (Y, Y'). Thus eventually we get

$$(W, W', H, H', X, X') \approx_{O(L\epsilon_2)} (U, W', H, H', X, X').$$

However, note that our construction is a two-source extractor itself. Thus, regardless of whether $\tilde{\alpha} \neq \tilde{\alpha}'$, we have that

$$(W, H, H', X, X') \approx_{O(L\epsilon_2)} (U, H, H', X, X').$$

We can cut the output length of the extractor to be $m = \Theta(\log(1/\epsilon_2))$ such that for any s in the support, we have $\Pr[U = s] = 2^{-m} = \sqrt{\epsilon_2}$. Thus we have for any s,

$$(H, H', X, X'|W = s) \approx_{O(L\sqrt{\epsilon_2})} (H, H', X, X'|U = s).$$

This means for any s,

$$(H, H', X, X'|W = s) \approx_{O(L\sqrt{\epsilon_2})} (H, H', X, X').$$

Let A be the event that $\tilde{\alpha} \neq \tilde{\alpha}'$. Note that $\Pr[A] \geq 1 - \epsilon_1$. Since A is determined by (H, H'), we have that for any s, $|\Pr[A|W = s] - \Pr[A]| \leq O(L\sqrt{\epsilon_2})$.

We now consider the probability distribution $(W'|W = s, A)$. This time we first condition on A. Note that conditioned on this event, we have

$$(W, W', H, H', X, X') \approx_{O(L\epsilon_2)} (U, W', H, H', X, X').$$

Thus again here we have that for any s,

$$(W', H, H', X, X'|W = s) \approx_{O(L\sqrt{\epsilon_2})} (W', H, H', X, X').$$

Therefore, we have for any s,

$$(W'|W = s, A) \approx_{O(L\sqrt{\epsilon_2})} (W'|A).$$

We can now bound the statistical distance between $(W'|W = s)$ and W'. We have

$$\begin{aligned}
&\left|(W'|W = s) - W'\right| \\
=&\left|(\Pr[A|W = s](W'|W = s, A) + \Pr[\bar{A}|W = s](W'|W = s, \bar{A})) - (\Pr[A](W'|A) + \Pr[\bar{A}](W'|\bar{A}))\right| \\
\leq&\left|\Pr[A]((W'|W = s, A) - W'|A)\right| + \left|(\Pr[A|W = s] - \Pr[A])(W'|W = s, A)\right| \\
&+ \left|\Pr[\bar{A}]((W'|W = s, \bar{A}) - (W'|\bar{A}))\right| + \left|(\Pr[\bar{A}|W = s] - \Pr[\bar{A}])(W'|W = s, \bar{A})\right| \\
\leq&\left|\Pr[A]((W'|W = s, A) - W'|A)\right| + \left|\Pr[\bar{A}]((W'|W = s, \bar{A}) - (W'|\bar{A}))\right| + O(L\sqrt{\epsilon_2}) \\
\leq&\left|((W'|W = s, A) - W'|A)\right| + \Pr[\bar{A}] \\
\leq&\epsilon_1 + O(L\sqrt{\epsilon_2}).
\end{aligned}$$

Note that the distribution of W' is a fixed probability distribution which is independent of s. Thus the construction gives a non-malleable code with error $\epsilon_1 + O(L\sqrt{\epsilon_2}) = \epsilon_1 + O(\log(1/\epsilon_1)\sqrt{\epsilon_2})$, and the rate of the code is $\Omega(\log(1/\epsilon_2)/n)$. ◀

We need to use the following simple inequality:

▶ **Fact 76.** *For any $0 < x \leq 1/3$, we have $1 - 3x \leq \frac{1-x}{1+x} < \frac{1+x}{1-x} \leq 1 + 3x$.*

We now have the following lemma, which gives a construction of non-malleable codes in the general case.

▶ **Lemma 77.** *Assume $\mathsf{2Ext} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^m$ satisfies the following conditions:*

- *It is a two-source extractor for entropy $n - \log(1/\epsilon')$ with error $\epsilon' \leq 2^{-(m+2)}$.*
- *It is a non-malleable two-source extractor for entropy $n - \log(1/\epsilon')$, which gives a non-malleable code in the two-split state model with error ϵ when either the tampering function f or the tampering function g has no fixed point.*

Then $\mathsf{2Ext}$ gives non-malleable code in the two-split state model with error $\epsilon + 2^{m+4}\epsilon'$.

Proof. Consider the tampering function $f : \{0,1\}^n \to \{0,1\}^n$ and $g : \{0,1\}^n \to \{0,1\}^n$. Let X and Y be two independent uniform distributions on $\{0,1\}^n$, let $p_0 = \Pr[f(X) = X]$, $q_0 = \Pr[g(Y) = Y]$, $p_1 = \Pr[f(X) \neq X] = 1 - p_0$ and $q_1 = \Pr[g(Y) \neq Y] = 1 - q_0$. Let the subsource of X conditioned on $f(X) = X$ be X_0, and the subsource of X conditioned on $f(X) \neq X$ be X_1. Thus $X = p_0X_0 + p_1X_1$. Similarly, we can define the subsources Y_0, Y_1 of Y such that $Y = q_0Y_0 + q_1Y_1$.

Consider the pairs of subsources (X_0, Y_0), (X_0, Y_1), (X_1, Y_0), and (X_1, Y_1), which have probability mass p_0q_0, p_0q_1, p_1q_0 and p_1q_1 respectively. Note that we have

$$(X, Y) = p_0q_0(X_0, Y_0) + p_0q_1(X_0, Y_1) + p_1q_0(X_1, Y_0) + p_1q_1(X_1, Y_1).$$

Let $W = \mathsf{2Ext}(X, Y)$. Consider any $s \in \{0,1\}^m$ and the uniform distribution on the pre-image of $W = s$ in (X, Y), call it Z_s. For any $i, j \in \{0,1\}$, let the subsource Z_{ij} stand for the uniform distribution on the pre-image of $W = s$ in (X_i, Y_j). Further let $r_{ij} = \Pr[\mathsf{2Ext}(X_i, Y_j) = s]$. Then we have

$$Z_s = \frac{\sum_{i,j} p_iq_jr_{ij}Z_{ij}}{\sum_{i,j} p_iq_jr_{ij}} = \sum_{i,j} \alpha_{ij}Z_{ij},$$

where $\alpha_{ij} = \frac{p_iq_jr_{ij}}{\sum_{i,j} p_iq_jr_{ij}}$.

We now have the following claim.

▷ Claim 78. For any $i, j \in \{0,1\}$, we have

- If either $p_i < \epsilon'$ or $q_j < \epsilon'$, then $\alpha_{ij} \leq 2^{m+1}\epsilon'$.
- Otherwise, $|\alpha_{ij}/(p_iq_j) - 1| \leq 2^{m+2}\epsilon'$

Proof of the claim. Note that $\sum_{i,j} p_iq_jr_{ij} = \Pr[W = s]$, and we have $\Pr[W = s] \geq 2^{-m} - \epsilon' > 2^{-(m+1)}$. Thus if either $p_i < \epsilon'$ or $q_j < \epsilon'$, we have

$$\alpha_{ij} = \frac{p_iq_jr_{ij}}{\sum_{i,j} p_iq_jr_{ij}} < 2^{m+1}\epsilon'.$$

Otherwise, both $p_i \geq \epsilon'$ and $q_j \geq \epsilon'$. This means that both X_i and Y_j have min-entropy at least $n - \log(1/\epsilon')$. Therefore we have $|r_{ij} - 2^{-m}| \leq \epsilon'$. Note that $\alpha_{ij}/(p_iq_j) = r_{ij}/\Pr[W = s]$ and we also have $.|\Pr[W = s] - 2^{-m}| \leq \epsilon'$. Since $\epsilon' \leq 2^{-(m+2)}$ by Fact 76 we have that

$$|\alpha_{ij}/(p_iq_j) - 1| \leq 2^{m+2}\epsilon'.$$ ◀

We now consider the distribution $\mathsf{2Ext}(T(Z_s))$, where for any distribution Z on $\{0,1\}^n \times \{0,1\}^n$, $T(Z)$ stands for the distribution $(f(x), g(y))$ where (x,y) is sampled from Z. Note that $\mathsf{2Ext}(Z_s)$ is fixed to s and and $\mathsf{2Ext}(T(Z_s))$ is the distribution of the decoded message after tampering. We have that $T(Z_s) = \sum_{i,j} \alpha_{ij} T(Z_{ij})$ and $\mathsf{2Ext}(T(Z_s)) = \sum_{i,j} \alpha_{ij} \mathsf{2Ext}(T(Z_{ij}))$. We will show that this distribution is close to the following distribution. For any $i,j \in \{0,1\}$ that are not both 0, if either $p_i < \epsilon'$ or $q_j < \epsilon'$, we define the distribution D_{ij} on $\{0,1\}^m$ to be a fixed constant (e.g., $\Pr[D_{ij} = 0^m] = 1$); otherwise since both X_i and Y_j have min-entropy at least $n - \log(1/\epsilon')$, $\mathsf{2Ext}$ gives a non-malleable code and thus $\mathsf{2Ext}(T(Z_{ij})$ is ϵ-close to a distribution D_{ij} independent of s. We let D_{00} be the distribution obtained by the identity function, i.e., for any s, D_{00} is fixed to be $I(s) = s$. We now claim that $\mathsf{2Ext}(T(Z_s))$ is close to the distribution $\sum_{i,j} p_i q_j D_{ij}$. We have

$$\left| \mathsf{2Ext}(T(Z_s)) - \sum_{i,j} p_i q_j D_{ij} \right| = \left| \sum_{i,j} \alpha_{ij} \mathsf{2Ext}(T(Z_{ij})) - \sum_{i,j} p_i q_j D_{ij} \right| \leq \sum_{i,j} \left| \alpha_{ij} \mathsf{2Ext}(T(Z_{ij})) - p_i q_j D_{ij} \right|.$$

For any $i,j \in \{0,1\}$, if either $p_i < \epsilon'$ or $q_j < \epsilon'$, we have the following bound.

$$|\alpha_{ij} \mathsf{2Ext}(T(Z_{ij})) - p_i q_j D_{ij}| \leq |\alpha_{ij} \mathsf{2Ext}(T(Z_{ij}))| + |p_i q_j D_{ij}| \leq 2^{m+1}\epsilon' + \epsilon' < 2^{m+2}\epsilon'.$$

Otherwise if i,j are not both 0 we have the following bound.

$$\begin{aligned} |\alpha_{ij} \mathsf{2Ext}(T(Z_{ij})) - p_i q_j D_{ij}| &\leq p_i q_j |\mathsf{2Ext}(T(Z_{ij})) - D_{ij}| + |(\alpha_{ij} - p_i q_j) \mathsf{2Ext}(T(Z_{ij}))| \\ &\leq p_i q_j \epsilon + 2^{m+2}\epsilon'. \end{aligned}$$

For the case of $i = j = 0$, we have that for any $(x,y) \in \mathsf{Supp}(Z_{00})$, $f(x) = x$ and $g(y) = y$. Thus $\mathsf{2Ext}(T(Z_{ij})) = s = D_{00}$ and we have

$$\begin{aligned} |\alpha_{ij} \mathsf{2Ext}(T(Z_{ij})) - p_i q_j D_{ij}| &\leq p_i q_j |\mathsf{2Ext}(T(Z_{ij})) - D_{ij}| + |(\alpha_{ij} - p_i q_j) \mathsf{2Ext}(T(Z_{ij}))| \\ &\leq 2^{m+2}\epsilon'. \end{aligned}$$

Therefore altogether we have

$$\left| \mathsf{2Ext}(T(Z_s)) - \sum_{i,j} p_i q_j D_{ij} \right| \leq \sum_{i,j} (p_i q_j \epsilon + 2^{m+2}\epsilon') = \epsilon + 2^{m+4}\epsilon'.$$

Since $\sum_{i,j} p_i q_j D_{ij}$ is obtained by $G(s)$ where G is a fixed probability distribution on the identity function and constant functions (the distribution of G only depends on f and g, but not on s), this implies that we have a non-malleable code in the 2 split-state model with error $\epsilon + 2^{m+4}\epsilon'$. ◀

We now have the following theorem.

▶ **Theorem 79.** *There are constants $0 < \eta, \mu < 1$ such that for any $n \in \mathcal{N}$ and $2^{-\frac{\mu n}{\log n}} \leq \epsilon \leq \eta$ there exists an explicit non-malleable code in the 2-split-state model with block length $2n$, rate $\Omega(\frac{\log\log\log(1/\epsilon)}{\log\log(1/\epsilon)})$ and error ϵ.*

Proof. We combine Theorem 75 and Lemma 77. Note that in Theorem 75, the construction is itself a two-source extractor for entropy $(1-\gamma)n$ with error $O(\log(1/\epsilon_1)\epsilon_2)$. To apply Theorem 75, we just need to ensure that

$$n \geq C \frac{\log\log(1/\epsilon_1)}{\log\log\log(1/\epsilon_1)} \log(n/\epsilon_2)$$

for some constant $C > 1$. We set $\epsilon_1 = \epsilon/2$ and $\epsilon_2 = 2^{-\Omega(\frac{\log\log\log(1/\epsilon)n}{\log\log(1/\epsilon)})}$. Note that

$$C\frac{\log\log(1/\epsilon_1)}{\log\log\log(1/\epsilon_1)}\log(n/\epsilon_2) = O(\frac{\log\log(1/\epsilon)}{\log\log\log(1/\epsilon)}\log n) + O(\frac{\log\log(1/\epsilon)}{\log\log\log(1/\epsilon)}\log(1/\epsilon_2)).$$

Since $2^{-\frac{\mu n}{\log n}} \leq \epsilon$ we have $\frac{\log\log(1/\epsilon)}{\log\log\log(1/\epsilon)}\log n = O(\frac{\log^2 n}{\log\log n})$. Thus we can set $\epsilon_2 = 2^{-\Omega(\frac{\log\log\log(1/\epsilon)n}{\log\log(1/\epsilon)})}$ to satisfy the inequality. Now we apply Lemma 77. We can set $\epsilon' = O(\log(1/\epsilon_1)\epsilon_2)$ since by Theorem 75 the construction is both a two-source extractor and a non-malleable two-source extractor for entropy $(1-\gamma)n$, and as long as $\epsilon \leq \eta$ for some appropriately chosen $\eta < 1$ we have $\log(1/\epsilon') \leq \gamma n$. Since in Theorem 75 we set the output of the extractor to be $m = \Theta(\log(1/\epsilon_2))$ such that $2^{-m} = \sqrt{\epsilon_2}$, we have that $\epsilon' \leq 2^{-(m+2)}$ and $2^{m+4}\epsilon' = O(\log(1/\epsilon_1)\sqrt{\epsilon_2})$. Thus by Lemma 77 the final error of the non-malleable code is

$$\epsilon_1 + O(\log(1/\epsilon_1)\sqrt{\epsilon_2}) + 2^{m+4}\epsilon' = \epsilon/2 + O(\log(1/\epsilon)\sqrt{\epsilon_2}).$$

Finally, notice that

$$\sqrt{\epsilon_2} = 2^{-\Omega(\frac{\log\log\log(1/\epsilon)n}{\log\log(1/\epsilon)})} \leq \alpha\frac{\epsilon}{\log(1/\epsilon)}$$

for any arbitrary constant $\alpha > 0$, since the latter is at least $\frac{1}{n}2^{-\frac{\mu n}{\log n}}$ and ϵ_2 is $2^{-\Omega(\frac{n\log\log n}{\log n})}$. Thus the final error of the non-malleable code is at most $\epsilon/2 + \epsilon/2 = \epsilon$, while the rate of the code, by Theorem 75, is $\Omega(\log(1/\epsilon_2)/n) = \Omega(\frac{\log\log\log(1/\epsilon)}{\log\log(1/\epsilon)})$. ◀

Next, we show how to achieve better error in the non-malleable two-source extractor and non-malleable codes. Recall that a bottleneck for error is the use of Reed-Solomon code in the construction. In order to get better error, we instead use a binary linear error correcting code and its generating matrix. It is easy to show using standard probabilistic argument that there exists a binary generating matrix that satisfies our requirements.

▶ **Theorem 80.** *There exists constants $0 < \alpha, \beta < 1$ such that for any $n \in \mathcal{N}$ there exists an $n \times m$ matrix over $\mathbf{F}_2$ with $n = \beta m$ which is the generating matrix of an asymptotically good code. Furthermore, Any sub-matrix formed by taking αn columns and the last $n/2$ rows has full column rank. In addition, for some $\epsilon = 2^{-O(n)}$, an ϵ-biased sample space over nm bits generates such a matrix with probability $1 - 2^{-\Omega(n)}$.*

Proof. We take an ϵ-biased sample space over nm bits for some $\epsilon = 2^{-O(n)}$. First, consider the sum of the rows over any non-empty subset of the rows. The sum is an m-bit string such that any non-empty parity is ϵ-close to uniform. Thus by the XOR lemma it is $2^{m/2}\epsilon$-close to uniform. We know a uniform m-bit string has weight $d = m/4$ with probability at least $1 - 2^{-\Omega(m)}$. Thus for this string the probability is at least $1 - 2^{-\Omega(m)} - 2^{m/2}\epsilon$. By a union bound the total failure probability is at most $2^n(2^{-\Omega(m)} + 2^{m/2}\epsilon) = 2^{-\Omega(n)}$ by an appropriate choice of β and $\epsilon = 2^{-O(n)}$.

Next, consider any sub-matrix formed by taking βm columns and the last $n/2$ rows, if it's truly uniform, then the probability that it has full column rank is at least $1 - \alpha n 2^{\alpha n - n/2} \geq 1 - 2^{-n/4}$ for $\alpha < 1/5$. Now by a union bound the total failure probability is at most

$$\binom{m}{\alpha n}(2^{-n/4} + \epsilon) \leq \left(\frac{em}{\alpha n}\right)^{\alpha n} 2^{-n/4+1} = \left(\frac{e}{\beta\alpha}\right)^{\alpha n} 2^{-n/4+1},$$

if we choose $\epsilon < 2^{-n/4}$. Note that for a fixed β, the quantity $(\frac{e}{\beta\alpha})^{\alpha}$ goes to 1 as α goes to 0. Thus we can choose α small enough such that this failure probability is also $2^{-\Omega(n)}$. Therefore altogether the failure probability is $2^{-\Omega(n)}$. ◀

Note that an ϵ-biased sample space over nm bits can be generated using $O(\log(nm/\epsilon)) = O(n)$ bits if $\epsilon = 2^{-O(n)}$. Now for any length $n \in \mathcal{N}$, we can compute the generating matrix (either using an ϵ-biased sample space or compute it deterministically in $2^{O(n)}$ time) once in the pre-processing step, and when we do encoding and decoding of the non-malleable code, all computation can be done in polynomial time.

Combining Theorem 67 and Theorem 55, we immediately obtain the following theorem.

▶ **Theorem 81.** *For any $n \in \mathcal{N}$ there exists a non-malleable code with efficient encoder-/decoder in the 2-split-state model with block length $2n$, rate $\Omega(\log\log n/\log n)$ and error $= 2^{-\Omega(n\log\log n/\log n)}$.*

8 Discussion and Open Problems

Several natural open problems remain here. The most intriguing one is how far we can push our new techniques. As mentioned above, one bottleneck here is that the computation of the merger is not a small space computation. If one can find a more succinct way to represent the computation, then it will certainly lead to further improvements (e.g., decrease the entropy requirement in two-source extractors to $O(\log n\sqrt{\log\log n})$). If in addition we can find a way to apply the recursive construction as in Nisan's generator [59], then it is potentially possible to decrease the entropy requirement in two-source extractors to $O(\log n \log\log\log n)$. We also believe our approach has the potential to eventually achieve truly optimal (up to constants) constructions. In addition, our techniques of treating the errors separately in non-malleable two-source extractors, may be useful in helping improve the rate of non-malleable codes for other classes of tampering functions (e.g., the affine tampering function and small depth circuits studied in [14]).

References

1 D. Aggarwal, Y. Dodis, T. Kazana, and M. Obremski. Non-malleable Reductions and Applications. In *Proceedings of the 47th Annual ACM Symposium on Theory of Computing*, 2015.

2 Divesh Aggarwal. Affine-evasive sets modulo a prime. Technical Report 2014/328, Cryptology ePrint Archive, 2014.

3 Divesh Aggarwal, Yevgeniy Dodis, and Shachar Lovett. Non-malleable Codes from Additive Combinatorics. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, 2014.

4 Boaz Barak, R. Impagliazzo, and Avi Wigderson. Extracting Randomness Using Few Independent Sources. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, pages 384–393, 2004.

5 Boaz Barak, Guy Kindler, Ronen Shaltiel, Benny Sudakov, and Avi Wigderson. Simulating Independence: New Constructions of Condensers, Ramsey Graphs, Dispersers, and Extractors. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, pages 1–10, 2005.

6 Boaz Barak, Anup Rao, Ronen Shaltiel, and Avi Wigderson. 2 Source Dispersers for $n^{o(1)}$ Entropy and Ramsey Graphs beating the Frankl-Wilson Construction. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, 2006.

7 Avraham Ben-Aroya, Gil Cohen, Dean Doron, and Amnon Ta-Shma. Two-Source Condensers with Low Error and Small Entropy Gap via Entropy-Resilient Functions. Technical Report TR18-066, ECCC, 2018.

8 Avraham Ben-Aroya, Dean Doron, and Amnon Ta-Shma. Explicit two-source extractors for near-logarithmic min-entropy. In *Proceedings of the 49th Annual ACM Symposium on Theory of Computing*, 2017.

9 Charles H. Bennett, Gilles Brassard, and Jean-Marc Robert. Privacy Amplification by Public Discussion. *SIAM Journal on Computing*, 17(2):210–229, April 1988.

10 Jean Bourgain. More on the sum-product phenomenon in prime fields and its applications. *International Journal of Number Theory*, 1:1–32, 2005.

11 N. Chandran, B. Kanukurthi, R. Ostrovsky, and L. Reyzin. Privacy amplification with asymptotically optimal entropy loss. In *Proceedings of the 42nd Annual ACM Symposium on Theory of Computing*, pages 785–794, 2010.

12 Eshan Chattopadhyay, Vipul Goyal, and Xin Li. Non-Malleable Extractors and Codes, with their Many Tampered Extensions. In *Proceedings of the 48th Annual ACM Symposium on Theory of Computing*, 2016.

13 Eshan Chattopadhyay and Xin Li. Explicit Non-Malleable Extractors, Multi-Source Extractors and Almost Optimal Privacy Amplification Protocols. In *Proceedings of the 57th Annual IEEE Symposium on Foundations of Computer Science*, 2016.

14 Eshan Chattopadhyay and Xin Li. Non-Malleable Codes and Extractors for Small-Depth Circuits, and Affine Functions. In *Proceedings of the 49th Annual ACM Symposium on Theory of Computing*, 2017.

15 Eshan Chattopadhyay and David Zuckerman. Non-malleable Codes against Constant Split-State Tampering. In *Proceedings of the 55th Annual IEEE Symposium on Foundations of Computer Science*, pages 306–315, 2014.

16 Eshan Chattopadhyay and David Zuckerman. Explicit Two-Source Extractors and Resilient Functions. In *Proceedings of the 48th Annual ACM Symposium on Theory of Computing*, 2016.

17 Mahdi Cheraghchi and Venkatesan Guruswami. Capacity of non-malleable codes. In *ITCS*, pages 155–168, 2014.

18 Mahdi Cheraghchi and Venkatesan Guruswami. Non-malleable Coding against Bit-Wise and Split-State Tampering. In *TCC*, pages 440–464, 2014.

19 Benny Chor and Oded Goldreich. Unbiased Bits from Sources of Weak Randomness and Probabilistic Communication Complexity. *SIAM Journal on Computing*, 17(2):230–261, 1988.

20 Gil Cohen. Local Correlation Breakers and Applications to Three-Source Extractors and Mergers. In *Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science*, 2015.

21 Gil Cohen. Making the Most of Advice: New Correlation Breakers and Their Applications. In *Proceedings of the 57th Annual IEEE Symposium on Foundations of Computer Science*, 2016.

22 Gil Cohen. Non-Malleable Extractors - New Tools and Improved Constructions. In *Proceedings of the 31st Annual IEEE Conference on Computational Complexity*, 2016.

23 Gil Cohen. Non-Malleable Extractors with Logarithmic Seeds. Technical Report TR16-030, ECCC, 2016.

24 Gil Cohen. Two-Source Extractors for Quasi-Logarithmic Min-Entropy and Improved Privacy Amplification Protocols. In *Proceedings of the 49th Annual ACM Symposium on Theory of Computing*, 2017.

25 Gil Cohen, Ran Raz, and Gil Segev. Non-Malleable Extractors with Short Seeds and Applications to Privacy Amplification. *SIAM Journal on Computing*, 43(2):450–476, 2014.

26 Gil Cohen and Leonard Schulman. Extractors for Near Logarithmic Min-Entropy. In *Proceedings of the 57th Annual IEEE Symposium on Foundations of Computer Science*, 2016.

27 Y. Dodis, J. Katz, L. Reyzin, and A. Smith. Robust Fuzzy Extractors and Authenticated Key Agreement from Close Secrets. In *Advances in Cryptology — CRYPTO '06, 26th Annual International Cryptology Conference, Proceedings*, pages 232–250, 2006.

28 Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM Journal on Computing*, 38:97–139, 2008.

29 Yevgeniy Dodis, Xin Li, Trevor D. Wooley, and David Zuckerman. Privacy Amplification and Non-Malleable Extractors Via Character Sums. *SIAM Journal on Computing*, 43(2):800–830, 2014.

30 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*, pages 601–610, 2009.

31 Dean Doron, Pooya Hatami, and William Hoza. Near-Optimal Pseudorandom Generators for Constant-Depth Read-Once Formulas. Technical Report TR18-183, ECCC, 2018.

32 Zeev Dvir, Swastik Kopparty, Shubhangi Saraf, and Madhu Sudan. Extensions to the Method of Multiplicities, with applications to Kakeya Sets and Mergers. In *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science*, 2009.

33 Zeev Dvir and Avi Wigderson. Kakeya sets, new mergers and old extractors. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science*, 2008.

34 Stefan Dziembowski, Tomasz Kazana, and Maciej Obremski. Non-malleable Codes from Two-Source Extractors. In *CRYPTO (2)*, pages 239–257, 2013.

35 Stefan Dziembowski and Krzysztof Pietrzak. Intrusion-Resilient Secret Sharing. In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '07, pages 227–237, Washington, DC, USA, 2007. IEEE Computer Society. doi:10.1109/FOCS.2007.35.

36 Stefan Dziembowski, Krzysztof Pietrzak, and Daniel Wichs. Non-Malleable Codes. In *ICS*, pages 434–452, 2010.

37 P. Erdős. Some remarks on the theory of graphs. *Bulletin of the American Mathematics Society*, 53:292–294, 1947.

38 Parikshit Gopalan, Raghu Meka, Omer Reingold, Luca Trevisan, and Salil Vadhan. Better pseudorandom generators from milder pseudorandom restrictions. In *Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science*, 2012.

39 Divya Gupta, Hemanta K. Maji, and Mingyuan Wang. Constant-rate Non-malleable Codes in the Split-state Model. Technical Report Report 2017/1048, Cryptology ePrint Archive, 2018.

40 Tom Gur and Igor Shinkar. An Entropy Lower Bound for Non-Malleable Extractors. Technical Report TR18-008, ECCC, 2018.

41 Venkatesan Guruswami, Christopher Umans, and Salil Vadhan. Unbalanced Expanders and Randomness Extractors from Parvaresh-Vardy Codes. *Journal of the ACM*, 56(4), 2009.

42 Yael Kalai, Xin Li, and Anup Rao. 2-Source Extractors Under Computational Assumptions and Cryptography with Defective Randomness. In *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 617–628, 2009.

43 Yael Tauman Kalai, Xin Li, Anup Rao, and David Zuckerman. Network Extractor Protocols. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 654–663, 2008.

44 B. Kanukurthi and L. Reyzin. Key agreement from close secrets over unsecured channels. In *EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, 2009.

45 Bhavana Kanukurthi, Lakshmibhavana Obbattu, and Sruthi Sekar. Four-state Non-malleable Codes with Explicit Constant Rate. In *Fifteenth IACR Theory of Cryptography Conference*, 2017.

46 Xin Li. Improved Constructions of Three Source Extractors. In *Proceedings of the 26th Annual IEEE Conference on Computational Complexity*, pages 126–136, 2011.

47 Xin Li. Design Extractors, Non-Malleable Condensers and Privacy Amplification. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing*, pages 837–854, 2012.

48 Xin Li. Non-Malleable Extractors, Two-Source Extractors and Privacy Amplification. In *Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science*, pages 688–697, 2012.

49 Xin Li. Extractors for a Constant Number of Independent Sources with Polylogarithmic Min-Entropy. In *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science*, pages 100–109, 2013.

50 Xin Li. New Independent Source Extractors with Exponential Improvement. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing*, pages 783–792, 2013.

51 Xin Li. Non-Malleable Condensers for Arbitrary Min-Entropy, and Almost Optimal Protocols for Privacy Amplification. In *12th IACR Theory of Cryptography Conference*, pages 502–531. Springer-Verlag, 2015. LNCS 9014.

52 Xin Li. Three Source Extractors for Polylogarithmic Min-Entropy. In *Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science*, 2015.

53 Xin Li. Improved Two-Source Extractors, and Affine Extractors for Polylogarithmic Entropy. In *Proceedings of the 57th Annual IEEE Symposium on Foundations of Computer Science*, 2016.

54 Xin Li. Improved Non-Malleable Extractors, Non-Malleable Codes and Independent Source Extractors. In *Proceedings of the 49th Annual ACM Symposium on Theory of Computing*, 2017.

55 C. J. Lu, Omer Reingold, Salil Vadhan, and Avi Wigderson. Extractors: Optimal up to Constant Factors. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing*, pages 602–611, 2003.

56 Ueli M. Maurer and Stefan Wolf. Privacy amplification secure against active adversaries. In *Advances in Cryptology — CRYPTO '97, 17th Annual International Cryptology Conference, Proceedings*, 1997.

57 Raghu Meka. Explicit resilient functions matching Ajtai-Linial. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms*, 2015.

58 Raghu Meka, Omer Reingold, and Avishay Tal. Pseudorandom Generators for Width-3 Branching Programs. Technical Report TR18-112, ECCC, 2018.

59 Noam Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12:449–461, 1992.

60 Noam Nisan and David Zuckerman. Randomness is Linear in Space. *Journal of Computer and System Sciences*, 52(1):43–52, 1996.

61 Anup Rao. Extractors for a Constant Number of Polynomially Small Min-entropy Independent Sources. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, 2006.

62 Ran Raz. Extractors with Weak Random Seeds. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, pages 11–20, 2005.

63 Renato Renner and Stefan Wolf. Unconditional authenticity and privacy from an arbitrarily weak secret. In *Advances in Cryptology — CRYPTO '03, 23rd Annual International Cryptology Conference, Proceedings*, pages 78–95, 2003.

64 Salil P. Vadhan. Constructing Locally Computable Extractors and Cryptosystems in the Bounded-Storage Model. *J. Cryptology*, 17(1):43–77, 2004. doi:10.1007/s00145-003-0237-x.

Optimal Separation and Strong Direct Sum for Randomized Query Complexity

Eric Blais
University of Waterloo, ON, Canada
eric.blais@uwaterloo.ca

Joshua Brody
Swarthmore College, PA, USA
brody@cs.swarthmore.edu

Abstract

We establish two results regarding the query complexity of bounded-error randomized algorithms.

Bounded-error separation theorem. There exists a total function $f : \{0,1\}^n \to \{0,1\}$ whose ϵ-error randomized query complexity satisfies $\overline{\mathrm{R}}_\varepsilon(f) = \Omega(\mathrm{R}(f) \cdot \log \frac{1}{\epsilon})$.

Strong direct sum theorem. For every function f and every $k \geq 2$, the randomized query complexity of computing k instances of f simultaneously satisfies $\overline{\mathrm{R}}_\varepsilon(f^k) = \Theta(k \cdot \overline{\mathrm{R}}_{\frac{\varepsilon}{k}}(f))$.

As a consequence of our two main results, we obtain an optimal superlinear direct-sum-type theorem for randomized query complexity: there exists a function f for which $\mathrm{R}(f^k) = \Theta(k \log k \cdot \mathrm{R}(f))$. This answers an open question of Drucker (2012). Combining this result with the query-to-communication complexity lifting theorem of Göös, Pitassi, and Watson (2017), this also shows that there is a total function whose public-coin randomized communication complexity satisfies $\mathrm{R}^{\mathrm{cc}}(f^k) = \Theta(k \log k \cdot \mathrm{R}^{\mathrm{cc}}(f))$, answering a question of Feder, Kushilevitz, Naor, and Nisan (1995).

2012 ACM Subject Classification Theory of computation → Probabilistic computation; Theory of computation → Oracles and decision trees

Keywords and phrases Decision trees, query complexity, communication complexity

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.29

Acknowledgements The first author thanks Alexander Belov and Shalev Ben-David for enlightening discussions and helpful suggestions. The second author thanks Peter Winkler for insightful discussions. Both authors wish to thank the anonymous referees for valuable feedback and for the reference to [26].

1 Introduction

We consider two fundamental questions related to the query complexity of functions in the bounded-error randomized setting: how the randomized query complexity of total functions scales with the allowable error ϵ (the *separation* problem), and how the query complexity of computing k instances of a function scales with the complexity of computing only 1 instance of the same function (the *direct sum* problem). Standard folklore arguments give upper bounds on how much the randomized query complexity can depend on ϵ and on k in these two problems; the results described below show that these well-known upper bounds are tight in general.

A randomized algorithm $\mathcal{A}$ *computes* a function $f : \mathcal{X}^n \to \{0,1\}$ over a finite set $\mathcal{X}^n$ *with error* $\epsilon \geq 0$ if for every input $x \in \mathcal{X}^n$, the algorithm outputs the value $f(x)$ with probability at least $1 - \epsilon$. The *query cost* of $\mathcal{A}$ is the maximum number of coordinates of x that it queries, with the maximum taken over both the choice of input x and the internal randomness of $\mathcal{A}$. The *ϵ-error (worst-case) randomized query complexity* of f (also known as the *randomized decision tree complexity* of f) is the minimum query complexity of an

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka; Article No. 29; pp. 29:1–29:17


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

algorithm $\mathcal{A}$ that computes f with error at most ϵ. We denote this complexity by $\mathrm{R}_\epsilon(f)$, and we write $\mathrm{R}(f) := \mathrm{R}_{\frac{1}{3}}(f)$ to denote the $\frac{1}{3}$-error randomized query complexity of f.

Another natural measure for the query cost of a randomized algorithm $\mathcal{A}$ is the *expected* number of coordinates of an input x that it queries. Taking the maximum expected number of coordinates queried by $\mathcal{A}$ over all inputs yields the *average query cost* of $\mathcal{A}$. The minimum average query complexity of an algorithm $\mathcal{A}$ that computes a function f with error at most ϵ is the *average ϵ-error query complexity* of f, which we denote by $\overline{\mathrm{R}}_\epsilon(f)$. We again write $\overline{\mathrm{R}}(f) := \overline{\mathrm{R}}_{\frac{1}{3}}(f)$. Note that $\overline{\mathrm{R}}_0(f)$ corresponds to the standard notion of *zero-error randomized query complexity* of f.

1.1 Our Results

Bounded-Error Separation Theorem for Query Complexity

One of the first tricks that one learns in the study of randomized algorithm is *success amplification*: it is possible to cheaply reduce the error of a randomized algorithm from $\frac{1}{3}$ to any $\varepsilon > 0$ by running the algorithm $O(\log \frac{1}{\varepsilon})$ times and outputting the most frequent answer. In the context of randomized query complexity, this means that for every function $f : \{0,1\}^n \to \{0,1\}$,

$$\mathrm{R}_\varepsilon(f) = O\big(\mathrm{R}(f) \cdot \log \tfrac{1}{\varepsilon}\big). \tag{1}$$

When considering partial functions, it is easy to see that the success amplification trick is optimal, as there are partial functions for which this relationship is tight (see Section 2.2). However, in the case of total functions, for many natural functions such as the majority function, parity function, dictator function, etc., the stronger bound $\mathrm{R}_\epsilon(f) = O\big(\mathrm{R}(f)\big)$ holds and until now it was not known whether the bound in (1) is tight for *any* total function. In fact, even separations between zero-error and $\frac{1}{3}$-error randomized query complexity were not known until very recently, when Ambainis et al. [2] showed that there exists a total function f for which $\overline{\mathrm{R}}_0(f) = \widetilde{\Omega}(\mathrm{R}(f)^2)$. Similarly, other separations between randomized query complexity and other measures of complexity have also only been established very recently [23, 1, 3, 4, 2].

In this work, we give the first separation within the bounded-error randomized query complexity setting. Our separation shows that the bound in (1) is optimal in general.

▶ **Theorem 1.** *For infinitely many values of n and every $2^{-(\frac{n}{\log n})^{1/3}} < \varepsilon \leq \frac{1}{3}$, there exists a total function $f : \{0,1\}^n \to \{0,1\}$ with randomized query complexity*

$$\overline{\mathrm{R}}_\varepsilon(f) = \Omega\big(\mathrm{R}(f) \cdot \log \tfrac{1}{\varepsilon}\big).$$

Note that by the trivial relation $\overline{\mathrm{R}}_\varepsilon(f) \leq \mathrm{R}_\varepsilon(f)$ between average and worst-case randomized query complexity, Theorem 1 implies the existence of a function f for which $\mathrm{R}_\varepsilon(f) \geq \Omega\big(\mathrm{R}(f) \cdot \log \frac{1}{\varepsilon}\big)$ and $\overline{\mathrm{R}}_\varepsilon(f) \geq \Omega\big(\overline{\mathrm{R}}(f) \cdot \log \frac{1}{\varepsilon}\big)$, giving optimal separations in both the worst-case randomized query complexity and average query complexity settings.

Strong Direct Sum Theorem

The *direct sum problem* asks how the cost of computing a function f scales with the number k of instances of the function that we need to compute. This problem has received a considerable amount of attention in the context of query complexity [18, 7, 24, 25, 19, 8, 13], communication complexity [20, 14, 11, 5, 21, 6], and beyond.

Given a function $f : \{0,1\}^n \to \{0,1\}$ and a parameter $k \geq 2$, define $f^k : \{0,1\}^{n \cdot k} \to \{0,1\}^k$ by setting $f^k(x^{(1)}, \ldots, x^{(k)}) = \big(f(x^{(1)}), \ldots, f(x^{(k)})\big)$. A simple union bound argument shows that the randomized query complexity of f^k is bounded above by

$$R_\epsilon(f^k) = O\big(k \cdot R_{\frac{\epsilon}{k}}(f)\big) \tag{2}$$

since we can run a randomized algorithm $\mathcal{A}$ that computes f with error at most $\frac{\epsilon}{k}$ on each of the k instances. An analogous upper bound holds in the average query complexity setting as well.

Jain, Klauck, and Santha [19] first considered the problem of showing a direct sum theorem for randomized query complexity. They showed that for every function f and for small enough constant $\delta > 0$, $\mathrm{R}_\epsilon(f^k) \geq \delta^2 k \cdot \mathrm{R}_{\frac{\epsilon}{1-\delta}+\delta}(f)$. Note that in this inequality, the allowable error on the right-hand side of the equation is *larger* than the ϵ error parameter, in contrast to the upper bound where it is (much) smaller. Ben-David and Kothari [8] obtained an improved direct sum theorem holds, showing that $\overline{\mathrm{R}}_\epsilon(f^k) \geq k \cdot \overline{\mathrm{R}}_\epsilon(f)$ holds for every function. This result is formally stronger since it implies the Jain–Klauck–Santha bound, but it also does not show that the error parameter on the right-hand-side of the inequality needs to be smaller than ϵ, as it is in the upper bound (2).

We show that the bound in (2) is tight in the average-case query complexity model.

▶ **Theorem 2.** *For every function $f : \{0,1\}^n \to \{0,1\}$, every $k \geq 2$, and every $0 \leq \epsilon \leq \frac{1}{20}$,*

$$\overline{\mathrm{R}}_\epsilon(f^k) = \Omega\left(k \cdot \overline{\mathrm{R}}_{\frac{\epsilon}{k}}(f)\right).$$

We establish Theorem 2 by proving a corresponding strong direct sum theorem in the distributional setting, as we discuss in more details in Section 1.3. It remains open to determine whether a similar strong direct sum theorem holds in the worst-case randomized query complexity model. However, in that setting Shaltiel [25] has shown that a proof of such a direct sum theorem *can't* be obtained via a corresponding theorem in the distributional setting, as a counterexample shows that direct sum theorems do not hold in this setting in general.

1.2 Applications

Superlinear Direct-Sum-Type Theorem for Query Complexity

Combining (1) and (2), we obtain a bound on the cost of computing k instances of a function f with bounded (constant) error and the cost of computing a single instance of the same function:

$$\mathrm{R}(f^k) = O\big(k \log k \cdot \mathrm{R}(f)\big). \tag{3}$$

Drucker [13, Open problem 2] asked if the superlinear dependence on k in (3) is necessary for any total function f. Theorems 1 and 2 give a positive answer to this question.

▶ **Corollary 3.** *There exists a total function $f : \{0,1\}^n \to \{0,1\}$ such that for all $1 \leq k \leq 2^{(\frac{n}{\log n})^{1/3}}$,*

$$\mathrm{R}(f^k) = \Theta\big(k \log k \cdot \mathrm{R}(f)\big).$$

Note that Corollary 3 stands in contrast to the quantum query complexity setting, where such a superlinear dependence on k is not required [10].

Superlinear Direct-Sum-Type Theorem for Communication Complexity

Let $\mathrm{R}^{\mathrm{cc}}(f)$ denote the minimum amount of communication required of a public-coin randomized protocol that computes a function $f : \{0,1\}^m \times \{0,1\}^n \to \{0,1\}$ with error at most $\frac{1}{3}$. As in the query complexity model, the communication complexity of the function f^k is bounded above by

$$\mathrm{R}^{\mathrm{cc}}(f^k) = O\big(k \log k \cdot \mathrm{R}^{\mathrm{cc}}(f)\big). \tag{4}$$

Feder, Kushilevitz, Naor, and Nisan [14] showed that this upper bound is not tight in general, as the equality function satisfies $\mathrm{R}^{\mathrm{cc}}(\mathrm{EQ}^k) = O\big(k \cdot \mathrm{R}^{\mathrm{cc}}(\mathrm{EQ})\big)$.[1] They then asked whether $\mathrm{R}^{\mathrm{cc}}(f^k) = O(k \cdot \mathrm{R}^{\mathrm{cc}}(f))$ holds for all functions or not [14, Open problem 2 in §7].

In the last few years, there has been much work on related direct sum questions. Molinaro, Woodruff, and Yaroslavtsev [21, 22] showed that in the *one-way* communication complexity model, the equality function does satisfy the superlinear direct sum bound $\mathrm{R}^{\mathrm{cc},\to}(\mathrm{EQ}^k) = \Theta\big(k \log k \cdot \mathrm{R}^{\mathrm{cc},\to}(\mathrm{EQ})\big)$. In the two-way communication complexity model that we consider, Barak, Braverman, Chen, and Rao [6] showed that every function f satisfies the direct sum $R(f^k) = \widetilde{\Omega}\big(\sqrt{k}\, R(f)\big)$, and this bound remains the state of the art as far as we know. Using the connection between information complexity and amortized communication complexity of Braverman and Rao [9], Ganor, Kol, and Raz [15] also showed that there is a partial function whose distributional communication complexity is exponentially larger than its amortized distributional communication complexity, showing that a tight direct sum theorem cannot hold in general in this setting. None of these results, however, answer Feder et al.'s original question.

Corollary 3 combined with the randomized query-to-communication lifting theorem of Göös, Pitassi, and Watson [17] answers Feder et al.'s question by showing that there is a function f for which the bound in (4) is tight.

▶ **Corollary 4.** *There is a constant $c > 0$ and a total function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ such that for all $1 \le k \le 2^{n^c}$,*

$$\mathrm{R}^{\mathrm{cc}}(f^k) = \Theta\big(k \log k \cdot \mathrm{R}^{\mathrm{cc}}(f)\big).$$

1.3 Proof Overviews

Bounded-Error Separation Theorem

The proof of Theorem 1 is established by following the general approach used to great effect by Ambainis et al. [2]: first, identify a partial function f for which the query complexity separation holds, then design a variant of the Göös–Pitassi–Watson (GPW) pointer function [16] that "embeds" the partial function into a total function and preserves the same separation.

The first step in this plan is accomplished by observing that the partial *gap identity* function $\textsc{GapID} : \{0,1\}^m \to \{0,1,*\}$ defined by

$$\textsc{GapID}(x) = \begin{cases} 1 & \text{if } |x| = 0, \\ 0 & \text{if } |x| = \lfloor \frac{m}{2} \rfloor, \\ * & \text{otherwise} \end{cases}$$

satisfies $\overline{\mathrm{R}}_{\varepsilon}(\textsc{GapID}) = \Theta\big(\mathrm{R}(\textsc{GapID}) \cdot \log \frac{1}{\varepsilon}\big)$ for every $\epsilon \ge 2^{-m}$.

[1] In fact, Feder et al. showed that the *private-coin* randomized communication complexity of EQ satisfies the stronger relation $\mathrm{R}^{\mathrm{cc,priv}}(\mathrm{EQ}^k) = o\big(k \cdot \mathrm{R}^{\mathrm{cc,priv}}(\mathrm{EQ})\big)$; their construction also directly establishes the result stated in the main text.

Ambainis et al. [2] also used (essentially) the same gap identity function to establish the separation $\overline{\mathrm{R}}_0(f) = \widetilde{\Omega}\big(\mathrm{R}(f)^2\big)$. In constructing a GPW pointer function analogue of the GapID function, however, Ambainis et al. lose a few logarithmic factors: their construction shows that there exists a total function $f : \{0,1\}^n \to \{0,1\}$ with ϵ-error randomized query complexity that satisfies $\mathrm{R}_\epsilon(f) = O(\sqrt{n}\log^2 n \log\frac{1}{\varepsilon})$ and $\mathrm{R}_\epsilon(f) = \Omega(\sqrt{n}\log\frac{1}{\epsilon})$. The polylogarithmic gap between those two bounds is not particularly important when comparing this query complexity to the zero-error randomized query complexity $\overline{\mathrm{R}}_0(f) = \widetilde{\Omega}(n)$ of the same function, but it makes it impossible to obtain any separation at all between $\mathrm{R}(f)$ and $\mathrm{R}_\varepsilon(f)$ whenever $\epsilon = \Omega(n^{-\log n})$. To prove Theorem 1, we need a new variant of the GPW pointer function whose analysis avoids *any* gap that is a non-constant function of n.

At a high-level, GPW pointer functions are constructed by defining an $n \times m$ array of cells, whose values are taken from some (typically fairly large) alphabet Σ. The first logarithmic gap in Ambainis et al.'s upper and lower bounds occurs because the upper bound is measured in terms of the number of *bits* queried by the algorithm while the lower bound is in terms of the number of *cells* queried by an algorithm. To eliminate this gap, we must either reduce the size of the alphabet from $|\Sigma| = O(\log n)$ to a constant size or modify the analysis so that both the upper and lower bounds are in terms of bit-query complexity. We do the latter, using the notion of *resilient functions* [12] to show that an algorithm must query a constant fraction of the bits of a cell to learn *anything* about the contents of that cell. Resilient functions were introduced by Chor et al. [12], who gave an essentially optimal construction using basic linear algebra and the probabilistic method. Sherstov recently created a gadget [26] resilient to approximate polynomial degree. This gadget is both similar in construction to [12] and in motivation to our work; it too removes some loss due to function inputs coming from large alphabets.

The second logarithmic gap in Ambainis et al.'s construction occurs because the location of the "special" cells that an algorithm seeks to discover in the GPW pointer function can be found by following a binary tree structure; the upper bound accounts for the $\log n$ cell queries an algorithm requires to follow this structure while the lower bound holds even if an algorithm finds these special cells in a single query. We bypass this problematic issue with a simple but powerful observation: in our setting, once we use resilient functions to encode the contents of each cell, there is no longer any requirement to keep the size $|\Sigma|$ of the alphabet for each cell in the GPW pointer function to be polylogarithmic in n and so we can include a *lot* more information in each cell without affecting the query complexity gap. We use this flexibility to replace pointers to the root of a binary tree structure with direct pointers to all the special cells in its leaves.

The details of the proof of Theorem 1 are presented in Section 2.

Strong Direct Sum Theorem

Our proof of the strong direct sum theorem proceeds by establishing an analogous result in the setting of distributional query complexity. The *ϵ-error distributional complexity* of $f : \{0,1\}^n \to \{0,1\}$ with respect to the distribution μ on $\{0,1\}^n$, denoted by $\mathrm{D}^\mu_\epsilon(f)$, is the minimum query complexity of a deterministic algorithm that computes the value $f(x)$ correctly with probability at least $1-\epsilon$ when x is drawn from μ.

The distributional complexity approach is also the one used in prior work on direct sum theorems for query complexity [19, 8]. The challenge with this approach, however, is that a strong direct sum theorem for distributional query complexity does *not* hold in general, as Shaltiel [25] demonstrated (see also §4 in [13]): there exists a function f and a distribution μ on f's domain for which $\mathrm{D}^{\mu^k}_\epsilon(f^k) = O\big(\epsilon k \mathrm{D}^\mu_\epsilon(f)\big)$.

A similar barrier to strong direct sum theorems exists in the communication complexity setting. Molinaro, Woodruff, and Yaroslavstev [21, 22] bypassed this barrier by considering randomized protocols that are allowed to abort with some bounded probability. They were then able to show that the information complexity of such communication protocols (in both the one-way and two-way communication models) satisfies a strong direct sum property.

Following an analogous approach, we consider randomized algorithms that are allowed to *abort* (or, equivalently, to output some value $\perp$ that corresponds to "don't know") with some probability at most δ. The *ϵ-error, δ-abort randomized query complexity* of a function f is denoted by $R_{\delta,\epsilon}(f)$. With a natural extension of Yao's minimax principle, we can obtain bounds on this randomized query complexity by considering the corresponding *ϵ-error, δ-abort distributional complexity* $\mathrm{D}^{\mu}_{\delta,\epsilon}(f)$ of a function f, which is the minimum query complexity of deterministic algorithms must err with probability at most ϵ and abort with probability at most δ when inputs are drawn from the distribution μ. We show that a strong direct sum theorem does hold in this setting.

▶ **Lemma 5.** *There exists a constant c such that for every function $f : \{0,1\}^n \to \{0,1\}$, every distribution μ on $\{0,1\}^n$, and every $0 \le \delta, \epsilon \le \frac{1}{40}$,*

$$\mathrm{D}^{\mu^k}_{\delta,\epsilon}(f^k) = \Omega\left(k \cdot \mathrm{D}^{\mu}_{\frac{1}{3},\frac{c\cdot\epsilon}{k}}(f)\right).$$

The proof of Theorem 2 is then obtained from this lemma by showing that an analogue of Yao's minimax principle holds for algorithms that can both err and abort. The full details of the proofs of Lemma 5 and Theorem 2 are presented in Section 3.

2 Bounded-Error Separation Theorem

We complete the proof of Theorem 1 in this section. In Section 2.1, we first define the pointer function PtrFcn at the heart of the proof. In Sections 2.2–2.4, we establish a lower bound on the query complexity of the PtrFcn function via reductions from the GapID function, and in Section 2.5, we provide a matching upper bound on this query complexity. We complete the proof of Theorem 1 in Section 2.6 by combining these results with the use of resilient functions.

2.1 Pointer Function

The total function at the heart of the proof of Theorem 1 is a variant of the Göös–Pitassi–Watson pointer function PtrFcn that we define below. Let $[n]$ denote the set $\{1,\dots,n\}$.

Define $\Gamma = \{0,1\} \times ([n] \cup \{\perp\})^m \times ([m] \cup \{\perp\})$ to be the set of symbols σ that encode a *value* that we denote by $\textsc{value}(\sigma)$, m *row pointers* that we denote by $\textsc{row}_1(\sigma),\dots,\textsc{row}_m(\sigma)$, and one *column pointer* that we denote $\textsc{col}(\sigma)$.

The function $\textsc{PtrFcn} : \Gamma^{n\times m} \to \{0,1\}$ is defined as follows. First, we represent an input $x \in \Gamma^{n\times m}$ as an $n \times m$ grid of *cells*. We say that a column $j^* \in [m]$ is *special* for x if $\textsc{value}(x_{i,j^*}) = 1$ for every $1 \le i \le n$. Then $\textsc{PtrFcn}(x) = 1$ if and only if

- There is a unique column j^* that is special for x;
- Within the special column j^*, there is a unique cell i^* called the *special cell*;
- $\textsc{row}_j(x_{i,j^*}) = \perp$ for all $i \ne i^*$ and all $j \ne j^*$;
- For all $j \ne j^*$, let $i_j := \textsc{row}_j(x_{i^*,j^*})$. Then, we have
  - $\textsc{value}(x_{i_j,j}) = 0$ (i.e., all cells pointed to by the special cell have value 0)
  - $|\{j \ne j^* : \textsc{col}(x_{i_j,j}) = j^* \wedge \textsc{row}_{j^*}(x_{i_j,j}) = i^*\}| = \lfloor \frac{m-1}{2} \rfloor$ (i.e., *half* the cells pointed to by the special cell point back to the special cell)

We call the cells (i_j, j) *linked cells*; linked cells that point back to the special cell are *good*. In summary, $\textsc{PtrFcn}(x) = 1$ if (i) there is a special column, (ii) within the special column, there is a special cell, (iii) all cells in the special column that are not the special cell have $\textsc{row}_j(x_{i,j*}) = \perp$ for all $j \neq j^*$, (iv) each linked cell has value 0, and (v) exactly half of the linked cells are good.

The following simple claim will be useful in obtaining the query complexity lower bound for $\textsc{PtrFcn}$.

▷ Claim 6. Let $\mathcal{A}$ be an ε-error randomized algorithm for $\textsc{PtrFcn}$. Let $z \in \textsc{PtrFcn}^{-1}(1)$, and let (i^*, j^*) be the special cell of z. Then $\mathcal{A}(z)$ probes (i^*, j^*) with probability at least $1 - 2\varepsilon$.

Proof. Let $\bar{z}$ be the same input as z except that $\textsc{value}(i^*, j^*) = 0$. Then $\textsc{PtrFcn}(z) \neq \textsc{PtrFcn}(\bar{z})$ but $z, \bar{z}$ differ only on the special cell. Whenever $\mathcal{A}$ doesn't probe the special cell, it must output the same value for z and $\bar{z}$, so it errs on either z or $\bar{z}$. By the error guarantee of $\mathcal{A}$ and a union bound, the probability that $\mathcal{A}$ doesn't probe cell (i^*, j^*) is at most 2ε. ◁

2.2 Lower Bound on the Query Complexity of GapID

We begin the proof of Theorem 1 by establishing a (simple, asymptotically optimal) lower bound on the average query complexity of the $\textsc{GapID}$ function.

▶ **Lemma 7.** *For every $m \geq 2$ and every $\epsilon < \frac{1}{2}$, $\overline{\mathrm{R}}_\epsilon(\textsc{GapID}) = \Omega\left(\min\{\log \frac{1}{\epsilon}, m\}\right)$.*

Proof. Fix any $\epsilon \geq 2^{-\frac{2}{3}m}$. We will show that $\overline{\mathrm{R}}_\epsilon(\textsc{GapID}) = \Omega\left(\log \frac{1}{\epsilon}\right)$. This suffices to complete the proof of the theorem since it implies that for any $\epsilon < 2^{-\frac{2}{3}m}$, $\overline{\mathrm{R}}_\epsilon(\textsc{GapID}) \geq \overline{\mathrm{R}}_{2^{-\frac{2}{3}m}}(\textsc{GapID}) = \Omega(m)$.

Let $\mathcal{A}$ be a randomized algorithm that computes $\textsc{GapID}$ with error probability at most ϵ. Let $Q \subseteq [m]$ be a random variable that denotes the set of coordinates queried by $\mathcal{A}$, and let $\xi := \xi(Q, x)$ denote the event that each coordinate of the input x queried by the algorithm has the value 0. Note that when the event $\xi(Q, x)$ occurs, $\mathcal{A}$ has the same behavior on input x as it does on the input 0^m. Since $\textsc{GapID}(0^m) = 1$ and $\mathcal{A}$ has error probability at most ϵ, this means that for every input $x \in \{0,1\}^m$,

$$\Pr[\mathcal{A}(x) = 0 \wedge \xi] = \Pr[\mathcal{A}(0^m) = 0 \wedge \xi] \leq \Pr[\mathcal{A}(0^m) = 0] \leq \epsilon$$

and so $\Pr[\mathcal{A}(x) = 1] \geq \Pr[\mathcal{A}(x) = 1 \wedge \xi] \geq \Pr[\xi] - \epsilon$.

Define μ to be the uniform distribution on all inputs $x \in \{0,1\}^m$ with $|x| = m/2$. To err with probability at most ϵ on those inputs, the algorithm $\mathcal{A}$ must satisfy $\Pr[\mathcal{A}(x) = 1] \leq \epsilon$ for every x in the support of μ. Combining this upper bound with the previous lower bound, we therefore have that

$$\Pr_{x\sim\mu,\, Q}[\xi] - \epsilon \leq \mathop{\mathrm{E}}_{x\sim\mu}\left[\Pr[\mathcal{A}(x) = 1]\right] \leq \epsilon \qquad \Longrightarrow \qquad \Pr_{x\sim\mu,\, Q}[\xi] \leq 2\epsilon. \tag{5}$$

For any value $1 \leq q \leq \frac{m}{3}$,

$$\Pr_{x\sim\mu,\, Q}\left[\xi \;\middle|\; |Q| = q\right] = \frac{\binom{m-q}{m/2}}{\binom{m}{m/2}} = \frac{\frac{m}{2}(\frac{m}{2}-1)\cdots(\frac{m}{2}-q+1)}{m(m-1)\cdots(m-q+1)}$$
$$> \left(\frac{\frac{m}{2}-q}{m-q}\right)^q > \left(\frac{1}{2} - \frac{q}{2(m-q)}\right)^q \geq 4^{-q}.$$

Therefore,

$$\Pr_{x\sim\mu,\,Q}\left[\xi \;\middle|\; |Q| \le \tfrac{1}{2}\log\tfrac{1}{4\epsilon}\right] > 4^{-\frac{1}{2}\log\frac{1}{4\epsilon}} = 4\epsilon.$$

Combining this inequality with (5), we obtain

$$2\epsilon \ge \Pr_{x\sim\mu,\,Q}[\xi] \ge \Pr\left[|Q| \le \tfrac{1}{2}\log\tfrac{1}{4\epsilon}\right] \cdot \Pr_{x\sim\mu,\,Q}\left[\xi \;\middle|\; |Q| \le \tfrac{1}{2}\log\tfrac{1}{4\epsilon}\right] > \Pr\left[|Q| \le \tfrac{1}{2}\log\tfrac{1}{4\epsilon}\right] \cdot 4\epsilon.$$

Rearranging the inequality yields $\Pr\left[|Q| \le \frac{1}{2}\log\frac{1}{4\epsilon}\right] < \frac{1}{2}$ and so the average query complexity of $\mathcal{A}$ is bounded below by

$$\mathrm{E}\left[|Q|\right] > \tfrac{1}{2}\log\tfrac{1}{4\epsilon} \cdot \Pr\left[|Q| > \tfrac{1}{2}\log\tfrac{1}{4\epsilon}\right] > \tfrac{1}{4}\log\tfrac{1}{4\epsilon}. \qquad \blacktriangleleft$$

2.3 Lower Bound on the Query Complexity of BlueRed

We wish to relate the average query complexity of PtrFcn to that of the GapID function. We do this by relating both query complexities to that of another partial function that we call BlueRed.

Let $\Sigma := \{\textsc{black}, \textsc{blue}, \textsc{red}\}$, and call a symbol *colored* if it is not black. The input is an $n \times m$ grid of entries from Σ, with the promise that each column contains a unique colored entry, and either all colored entries are red, or half the colored entries are blue. Formally, we define $\textsc{BlueRed} : \Sigma^{n\times m} \to \{0, 1, *\}$ as follows:

$$\textsc{BlueRed}(x) = \begin{cases} 1 & \text{if each column has 1 colored entry \& all colored entries are } \textsc{red}, \\ 0 & \text{if each column has 1 colored entry \& exactly } \lfloor \tfrac{m}{2} \rfloor \text{ entries are } \textsc{blue}, \\ * & \text{otherwise.} \end{cases}$$

The following reduction shows that the average query complexity of BlueRed is $\Theta(n)$ times as large as that of the GapID function.

▶ **Lemma 8.** *For every $\epsilon > 0$, $\overline{\mathrm{R}}_\varepsilon(\textsc{BlueRed}) \ge \frac{n}{4} \cdot \overline{\mathrm{R}}_\varepsilon(\textsc{GapID})$.*

Proof. Fix any algorithm $\mathcal{A}$ that computes BlueRed with error at most ϵ and has expected query cost $c = \overline{\mathrm{R}}_\varepsilon(\textsc{BlueRed})$. We will use $\mathcal{A}$ to construct an algorithm $\mathcal{B}$ that computes GapID with error at most ϵ and expected cost $4c/n$.

Given an input $x \in \{0,1\}^m$, the algorithm $\mathcal{B}$ constructs an instance of the BlueRed problem in the following way. First, it generates indices $i_1, \ldots, i_m \in [n]$ independently and uniformly at random. Then it defines

$$y_{i,j} = \begin{cases} \textsc{red} & \text{if } i = i_j \text{ and } x_j = 0, \\ \textsc{blue} & \text{if } i = i_j \text{ and } x_j = 1, \\ \textsc{black} & \text{if } i \ne i_j. \end{cases}$$

Finally, the algorithm $\mathcal{B}$ emulates the algorithm $\mathcal{A}$ on input y, querying the value of x_j whenever $\mathcal{A}$ queries the bit (i_j, j) for some $j \le m$. This construction guarantees that $\mathcal{B}$ computes GapID with error at most ϵ; its query complexity corresponds to the number of red or blue entries that are queried by $\mathcal{A}$.

Let $Q \subseteq [n] \times [m]$ be the random variable that denotes the set of indices queried by $\mathcal{A}$, and let $C \subseteq [m]$ denote the set of columns whose red or blue entry is queried by $\mathcal{A}$. Without loss of generality, we may assume that $\mathcal{A}$ does not query any entry of a column

after it finds the colored entry within that column. We partition C into two sets C_{early} and C_{late}, where C_{early} denotes the set of columns whose colored entry is found within the first $\frac{n}{2}$ queries to that column and C_{late} denotes the set of columns whose colored entry was found with more than $\frac{n}{2}$ queries to that column. Let $X_1, X_2, \ldots, X_{|Q|}$ be indicator variables where $X_k = 1$ if and only if the kth query (i, j) made by $\mathcal{A}$ is RED or BLUE *and* is one of the first $\frac{n}{2}$ queries to column j. Since each value i_j is drawn uniformly at random from $[n]$, each of these indicator variables has expected value $\mathrm{E}[X_k] \leq \frac{2}{n}$. Therefore,

$$\mathrm{E}\left[|C_{\text{early}}|\right] = \mathrm{E}\left[\sum_{i \leq |Q|} X_i\right] \leq \frac{2}{n}\,\mathrm{E}\left[|Q|\right].$$

Furthermore, by definition at least $\frac{n}{2}$ queries are made to each column in C_{late} so the expected size of this set is bounded by $\mathrm{E}\left[|C_{\text{late}}|\right] \leq \frac{2}{n}\,\mathrm{E}\left[|Q|\right]$ and

$$\mathrm{E}\left[|C|\right] = \mathrm{E}\left[|C_{\text{early}}|\right] + \mathrm{E}\left[|C_{\text{late}}|\right] \leq \frac{4}{n}\,\mathrm{E}\left[|Q|\right].$$

Thus, the expected query cost of $\mathcal{B}$ is at most $\frac{4}{n} \cdot \overline{\mathrm{R}}_{\varepsilon}(\textsc{BlueRed})$, as we wanted to show. ◀

2.4 Lower Bound on the Query Complexity of PtrFcn

▶ **Lemma 9.** *For every* $0 \leq \epsilon \leq \frac{1}{4}$, $\overline{\mathrm{R}}_{\epsilon}(\textsc{PtrFcn}) \geq \overline{\mathrm{R}}_{2\epsilon}(\textsc{BlueRed})$.

Proof. Let $\mathcal{A}$ be a randomized algorithm that computes PTRFCN with error at most ϵ and expected query cost $q := \overline{\mathrm{R}}_{\epsilon}(\textsc{PtrFcn})$. We use $\mathcal{A}$ to construct a randomized algorithm $\mathcal{B}$ that computes BLUERED with the same cost and error at most 2ϵ.

Let x be an input for BLUERED. Each time $\mathcal{A}$ queries a cell, $\mathcal{B}$ queries the corresponding entry in x. If the entry in x is BLACK, then $\mathcal{B}$ returns $\langle 1, \perp, \ldots, \perp \rangle$. If the entry in x is RED, then $\mathcal{B}$ returns $\langle 0, \perp, \ldots, \perp \rangle$. Finally, if the entry of x is BLUE, then $\mathcal{B}$ terminates the emulation and returns 0. If $\mathcal{A}$ reaches the end of the emulation without having been terminated, $\mathcal{B}$ outputs the same result as $\mathcal{A}$.

The query complexity of $\mathcal{B}$ is at most that of $\mathcal{A}$. It remains to show that $\mathcal{B}$ errs with probability at most 2ϵ. There are two cases to consider.

The first case is when $x \in \textsc{BlueRed}^{-1}(1)$. Then x maps directly to an input $z \in \textsc{PtrFcn}^{-1}(0)$ and hence $\mathcal{B}$ errs with probability at most ϵ on x.

The second case is when $x \in \textsc{BlueRed}^{-1}(0)$. Let z be an arbitrary 1-input for PTRFCN such that (i) $z_{i,j} = \langle 1, \perp, \ldots, \perp \rangle$ whenever $x_{i,j} =$ BLACK, (ii) $z_{i,j} = \langle 0, \perp, \ldots, \perp \rangle$ whenever $x_{i,j} =$ RED, and (iii) the special entry and good entries of z correspond to BLUE entries of x. It might not be possible to completely emulate $\mathcal{A}$ on input z without knowing the exact set of BLUE entries. However, $\mathcal{B}$ doesn't need to fully emulate $\mathcal{A}$ – it only needs to know how to map BLACK and RED entries. Once a BLUE entry is probed, $\mathcal{B}$ halts and outputs 0. In this way, we claim that $\mathcal{B}$ on input x probes the same cells as $\mathcal{A}$ on input z until it halts. Therefore its output is the same as $\mathcal{A}(z)$ unless $\mathcal{A}(z)$ probes the special cell or a good cell. Moreover, in this case, $\mathcal{B}$ outputs correctly with certainty. Thus, by Claim 6, the error of $\mathcal{B}$ is at most

$$\Pr[\mathcal{B} \text{ errs}] \leq \Pr[\mathcal{B} \text{ probes no blue cells}] \leq \Pr[\mathcal{A} \text{ doesn't probe special cell}] \leq 2\varepsilon\ .$$ ◀

2.5 Upper Bound on the Query Complexity of PtrFcn

The proof of Theorem 1 also requires a tight upper bound on the (worst-case) randomized query complexity of PTRFCN. This argument is straightforward, and similar to the analysis of Ambainis et al. [2] for their analogous pointer function.

Algorithm 1: PtrFcnSolver(x).

$S \leftarrow [m]$;
$T \leftarrow$ a random subset of $[m]$ of size $|T| = \log \frac{1}{\epsilon}$;
for *each cell* (i, j) *in a column in* T **do**
  if $\textsc{value}(x_{i,j}) = 0 \wedge \textsc{col}(x_{i,j}) \in S$ **then**
    $j^* \leftarrow \textsc{col}(x_{i,j})$;
    $i^* \leftarrow \textsc{row}_{j^*}(x_{i,j})$;
    $valid \leftarrow \textsc{True}$;
    while $|S| > 1 \wedge valid$ **do**
      $\ell \leftarrow$ any column in $S \setminus \{j^*\}$;
      if $\textsc{value}(x_{\textsc{row}_\ell(x_{i^*,j^*}),\ell}) = 0$ **then**
        $S \leftarrow S \setminus \{\ell\}$;
      else
        $valid \leftarrow \textsc{False}$;
    if $|S| = 1$ **then**
      break;
if $|S| = 1$ **then**
  fix $j \in S$. **return** 1 if (i) column j is special, (ii) there is a special cell within column j, (iii) all cells linked by the special cell have value 0, and (iv) half of linked cells point back to the special cell.
return 0

▶ **Lemma 10.** $\mathrm{R}_\varepsilon(\textsc{PtrFcn}) = O(n \log \frac{1}{\epsilon} + m)$.

Proof. The algorithm that computes the PtrFcn function is described in Algorithm 1. In this algorithm, the set S corresponds to the set of potential special columns. The query complexity of PtrFcnSolver follows from the fact that each iteration of the inner while loop either eliminates one of the columns from the set S of candidates or one of the $n \log \frac{1}{\epsilon}$ cells in the columns in T. The final check of the (lone remaining) potential special column at the end of the algorithm examines at most $n + m$ cells.

Whenever the PtrFcnSolver returns the value 1, then it in fact has observed a certificate that $\textsc{PtrFcn}(x) = 1$ so the algorithm has perfect soundness.

Conversely, suppose $\textsc{PtrFcn}(x) = 1$. Exactly half of the columns are good, so T contains such a cell with probability at least $1 - (1/2)^{\log(1/\varepsilon)} = 1 - \varepsilon$. Now, consider the for loop iteration when the first good cell (i, j) is selected. Since (i, j) is a good cell, it points back to the special cell, which in turn points to a linked cell in all columns except the special column. For any remaining $j' \neq j \in S$, $\textsc{PtrFcn}(x)$ probes the linked cell in column j', verifies the value equals 0, and removes it from S. In this way, the remaining columns in S save the special column are eliminated. Once we reduce S to a single remaining candidate, we can probe all cells in this column and all linked cells using $n + m$ queries to verify that indeed $\textsc{PtrFcn}(x) = 1$. ◀

2.6 Completing the Proof of Theorem 1

The last ingredient that we need to complete the proof of Theorem 1 is the concept of *resilient functions* [12].

▶ **Definition 11.** *The function* $\phi : \{0,1\}^n \to \{0,1\}^m$ *is* t-resilient *for some* $1 \leq t < n$ *if for any set* $S \subseteq [n]$ *of* $|S| \leq t$ *coordinates and any assignment of values for the inputs* $\{x_i\}_{i \in S}$*, when the values* $\{x_i\}_{i \in [n] \setminus S}$ *are set uniformly at random then* $\phi(x)$ *is uniformly distributed in* $\{0,1\}^m$.

We use the following existence result on resilient functions that was established by Chor et al. [12].

▶ **Theorem 12** (Chor et al. [12]). *For every large enough n, there is a function $\phi : \{0,1\}^n \to \{0,1\}^m$ that is $\frac{n}{3}$-resilient and satisfies $m \geq 0.08n$.*

We use resilient functions to bound the query complexity of functions via the following lemma.

▶ **Lemma 13.** *Fix a finite set $\mathcal{X}$ of cardinality $|\mathcal{X}| = 2^\ell$ for some integer $\ell \geq 1$ and let $\phi : \{0,1\}^N \to \mathcal{X}$ be an $\frac{N}{3}$-resilient function. Then for every function $f : \mathcal{X}^m \to \{0,1\}$ and every $\epsilon \geq 0$,*

$$\mathrm{R}_\epsilon(f \circ \phi) = \Theta(N \cdot \mathrm{R}_\epsilon(f)) \qquad \text{and} \qquad \overline{\mathrm{R}}_\epsilon(f \circ \phi) = \Theta(N \cdot \overline{\mathrm{R}}_\epsilon(f)).$$

Proof. The upper bounds follow immediately from the observation that if $\mathcal{A}$ is a randomized algorithm that computes f with ϵ-error, then we can define a algorithm $\mathcal{B}$ that computes $f \circ \phi$ with the same error by simulating $\mathcal{A}$ and querying the N bits to observe the value $\phi(x)$ to return to each query.

For the lower bounds, let $\mathcal{A}$ be a randomized algorithm that computes $f \circ \phi$ with error at most ϵ. We define an algorithm $\mathcal{B}$ for computing f that simulates $\mathcal{A}$ in the following way. For the first $\frac{N}{3}$ queries to a cell, $\mathcal{B}$ answers the queries with uniformly random variables in $\{0,1\}$. On a query to the $(\frac{N}{3}+1)$-th bit of a cell, $\mathcal{B}$ queries the value v of the corresponding cell in x. It then draws a value z in $\phi^{-1}(v)$ uniformly at random among all values that agree with the $\frac{N}{3}$ bits output so far. The current query and all further queries to bits of that cell are then answered using z. Once $\mathcal{A}$ terminates, $\mathcal{B}$ returns $\mathcal{A}$'s output and terminates as well.

The correctness of $\mathcal{B}$ follows directly from the correctness of $\mathcal{A}$. Furthermore, on any input for which $\mathcal{A}$ makes q queries, $\mathcal{B}$ makes at most $q/(N/3)$ queries since $N/3$ distinct queries of $\mathcal{A}$ are required for each query that $\mathcal{B}$ eventually makes to x. Thus both the average-case and worst-case query complexities of $\mathcal{B}$ are bounded by $3/N$ times the corresponding query complexities of $\mathcal{A}$. ◀

We are now ready to complete the proof of the separation theorem.

Proof of Theorem 1. Fix $m = n = 2^\ell - 1$ for any integer $\ell \geq 1$ so that $|\Gamma| = 2^{\ell(2^\ell-1)+\ell+1}$ is a power of 2. Fix a $\frac{C}{3}$-resilient function $\phi : \{0,1\}^C \to \Gamma$ for some $C \leq 12.5 \log |\Gamma|$ and define the function $\textsc{EncFcn} = \textsc{PtrFcn} \circ \phi$. By Lemmas 13 and 10,

$$\mathrm{R}_\varepsilon(\textsc{EncFcn}) = O\big(C(n \log \tfrac{1}{\epsilon} + m)\big) = O\big(Cn \log \tfrac{1}{\epsilon}\big).$$

In particular, setting $\epsilon = \frac{1}{3}$ we obtain $\mathrm{R}(\textsc{EncFcn}) = O(Cn)$.

Using Lemma 13, 9, and 8, we obtain the chain of inequalities

$$\overline{\mathrm{R}}_\varepsilon(\textsc{EncFcn}) = \Omega\big(C \cdot \overline{\mathrm{R}}_\epsilon(\textsc{PtrFcn})\big) = \Omega\big(C \cdot \overline{\mathrm{R}}_{2\epsilon}(\textsc{BlueRed})\big) = \Omega\big(Cn \cdot \overline{\mathrm{R}}_{2\varepsilon}(\textsc{GapID})\big).$$

By Lemma 7, when $\epsilon > 2^{-m} = 2^{-n}$ this implies that

$$\overline{\mathrm{R}}_\varepsilon(\textsc{EncFcn}) = \Omega\big(Cn \log \tfrac{1}{\epsilon}\big) = \Omega\big(\log \tfrac{1}{\epsilon} \cdot \mathrm{R}(\textsc{EncFcn})\big).$$

Theorem 1 is obtained by noting that $\textsc{EncFcn}$ is a function on $N = O(mn|\Gamma|) = O(n^3 \log n)$ variables. ◀

3 Strong Direct Sum Theorem

We establish Theorem 2 by proving a corresponding direct sum result in the distributional model and applying a Yao minimax principle for algorithms that err and abort with bounded probability.

We introduce the model of algorithms that can abort in Section 3.1, where we also relate this model to the average query complexity setting of randomized algorithms and establish a Yao minimax principle. In Section 3.2, we establish the main technical result, a strong direct sum theorem for distributional complexity. We complete the proof of Theorem 2 itself in Section 3.3 and the proofs of Corollaries 3 and 4 are completed in Section 3.4.

3.1 Algorithms That Can Abort

We consider randomized algorithms that are allowed to *err* and *abort*. In this setting, an algorithm outputs $\perp$ instead of giving a valid output when it chooses to abort. Let $\mathrm{D}^{\mu}_{\delta,\varepsilon}(f)$ and $\mathrm{R}_{\delta,\varepsilon}(f)$ denote the distributional and randomized query complexities of f when the algorithm aborts with probablity at most δ and errs with probability at most ε.

Randomized query complexity in the setting where algorithms can abort with constant probability δ is asymptotically equivalent to the average randomized query complexity.

▶ **Proposition 14.** *For every function $f : \{0,1\}^n \to \{0,1\}$, every $0 \le \epsilon < \frac{1}{2}$ and every $0 < \delta < 1$,*

$$\delta \cdot \mathrm{R}_{\delta,\varepsilon}(f) \le \overline{\mathrm{R}}_{\epsilon}(f) \le \tfrac{1}{1-\delta} \cdot \mathrm{R}_{\delta,(1-\delta)\epsilon}(f).$$

Proof. For the first inequality, let $\mathcal{A}$ be a randomized algorithm that computes f with ϵ error and has expected query complexity q. Let $\mathcal{B}$ be the randomized algorithm $\mathcal{B}$ that simulates $\mathcal{A}$ except that whenever $\mathcal{A}$ tries to make more than q/δ queries, it aborts. The algorithm $\mathcal{B}$ also computes f with error at most ϵ, and it has worst-case query complexity q/δ. Furthermore, by Markov's inequality, $\mathcal{B}$ aborts with probability at most δ.

For the second inequality, let $\mathcal{B}$ be a randomized algorithm with query complexity q that computes f with error probability at most $(1-\delta)\epsilon$ and abort probability at most δ. Let $\mathcal{A}$ be the randomized algorithm that simulates $\mathcal{B}$ until that algorithm does not abort, then outputs the same value. The error probability of $\mathcal{B}$ conditioned on it not aborting is at most $\frac{(1-\delta)\epsilon}{1-\delta} = \epsilon$, so the algorithm $\mathcal{A}$ also errs with probability at most ϵ, and its expected query complexity is $q(1+\delta+\delta^2+\cdots) = \frac{q}{1-\delta}$. ◀

Yao's minimax principle can be adapted for the setting of algorithms that abort as follows.

▶ **Lemma 15.** *For any $\alpha, \beta > 0$ such that $\alpha + \beta \le 1$, we have*

$$\max_{\mu} \mathrm{D}^{\mu}_{\delta/\alpha,\varepsilon/\beta}(f) \le \mathrm{R}_{\delta,\varepsilon}(f) \le \max_{\mu} \mathrm{D}^{\mu}_{\alpha\delta,\beta\varepsilon}(f).$$

Proof. We handle the initial inequality (i.e., the *easy direction*) first. Fix a q-query randomized algorithm $\mathcal{A}$ achieving $\mathrm{R}_{\delta,\varepsilon}(f)$. By the guarantee of $\mathcal{A}$, we have that for any input x, $\mathcal{A}$ aborts with probability at most δ and errs with probabiltiy at most ε. Let $\mathbf{1}_{\delta}(x)$ and $\mathbf{1}_{\varepsilon}(x)$ be indicator variables for the events that $\mathcal{A}$ aborts on x and $\mathcal{A}$ errs on x respectively. Then, we have $\mathrm{E}_R[\mathbf{1}_{\delta}(x)] \le \delta$ and similarly $\mathrm{E}_R[\mathbf{1}_{\varepsilon}(x)] \le \varepsilon$ when the expectation is taken over the randomness R of the algorithm $\mathcal{A}$. Next, fix any input distribution μ and let $X \sim \mu$. It follows that

$$\mathop{\mathrm{E}}_{R}\left[\mathop{\mathrm{E}}_{X}[\mathbf{1}_{\delta}(X)]\right] = \mathop{\mathrm{E}}_{X}\left[\mathop{\mathrm{E}}_{R}[\mathbf{1}_{\delta}(X)]\right] \le \delta \quad \text{and} \quad \mathop{\mathrm{E}}_{R}\left[\mathop{\mathrm{E}}_{X}[\mathbf{1}_{\varepsilon}(X)]\right] = \mathop{\mathrm{E}}_{X}\left[\mathop{\mathrm{E}}_{R}[\mathbf{1}_{\varepsilon}(X)]\right] \le \varepsilon.$$

Using Markov's inequality twice, we have

$$\Pr_R\left[\mathrm{E}_X[\mathbf{1}_\delta(X)] > \delta/\alpha\right] < \alpha \qquad \text{and} \qquad \Pr_R\left[\mathrm{E}_X[\mathbf{1}_\varepsilon(X)] > \varepsilon/\beta\right] < \beta.$$

By a union bound, there exists a setting of the random string R such that both $\mathrm{E}[\mathbf{1}_\delta(X)] \le \delta/\alpha$ and $\mathrm{E}[\mathbf{1}_\varepsilon(X)] \le \varepsilon/\beta$. Fixing this R gives a q-query deterministic algorithm that aborts with probability at most δ/α and errs with probability at most ε/β, hence $\mathrm{D}^\mu_{\delta/\alpha,\varepsilon/\beta}(f) \le \mathrm{R}_{\delta,\varepsilon}(f)$.

For the second inequality, let $c := \max_\mu \mathrm{D}^\mu_{\alpha\delta,\beta\varepsilon}(f)$. Consider a two-player, zero-sum game where player 1 selects a c-query deterministic algortihm $\mathcal{A}$ for f, player 2 selects an input x, and player 1 is paid $-\varepsilon$ if $\mathcal{A}(x)$ aborts, $-\delta$ if $\mathcal{A}(x)$ errs, and 0 otherwise. Note that each mixed strategy for player 1 corresponds to a randomized algorithm and each mixed strategy for player 2 corresponds to an input distribution μ. By our choice of c, it follows that for any mixed strategy for player 2, player 1 can obtain payoff $-\varepsilon(\alpha\delta) - \delta(\beta\varepsilon) \ge -\varepsilon\delta$. By the minimax theorem, it follows that there is a mixed strategy for player 1 (i.e., a c-query randomized algorithm $\mathcal{A}$) that provides the same payoff for every choice of player 2. Finally, note that $\mathcal{A}$ aborts with probability at most δ and errs with probability at most ε; otherwise, the payoff would be less than $-\varepsilon\delta \le -\varepsilon\delta(\alpha+\beta)$. We've shown a c-query randomized algorithm that aborts w/probability at most δ and errs w/probability at most ε, hence $\mathrm{R}_{\delta,\varepsilon}(f) \le c = \max_\mu \mathrm{D}^\mu_{\alpha\delta,\beta\varepsilon}(f)$. ◀

3.2 Strong Direct Sum for Distributional Complexity

We prove a slightly more precise variant of Lemma 5.

▶ **Lemma 16.** *For every function $f : \{0,1\}^n \to \{0,1\}$, every distribution μ on $\{0,1\}^n$, and every $0 \le \delta, \epsilon \le \frac{1}{4}$,*

$$\mathrm{D}^{\mu^k}_{\delta,\epsilon}(f^k) = \Omega\left(k \cdot \mathrm{D}^\mu_{\frac{1}{10}+4\delta+4\epsilon,\frac{48\epsilon}{k}}(f)\right).$$

Proof. Let $\mathcal{A}$ be a deterministic algorithm with query complexity q that computes f^k with error probability at most ϵ and abort probability at most δ when the input $x = (x^{(1)},\dots,x^{(k)})$ is drawn from μ^k. Then conditioned on $\mathcal{A}$ not aborting, it outputs the correct value of f^k with probability at least $1 - \frac{\epsilon}{1-\delta} \ge 1 - 2\epsilon$ and

$$\begin{aligned} 1 - 2\epsilon &\le \Pr_{x\sim\mu^k}\left[\mathcal{A}(x) = f^k(x) \;\middle|\; \mathcal{A}(x) \ne \perp\right] \\ &= \prod_{i\le k} \Pr_{x\sim\mu^k}\left[\mathcal{A}(x)_i = f(x^{(i)}) \;\middle|\; \mathcal{A}(x)_{<i} = f^k(x)_{<i}, \mathcal{A}(x) \ne \perp\right]. \end{aligned}$$

This implies that at least $\frac{2}{3}k$ indices $i \in [k]$ satisfy

$$\Pr_{x\sim\mu^k}\left[\mathcal{A}(x)_i \ne f(x^{(i)}) \;\middle|\; \mathcal{A}(x)_{<i} = f^k(x)_{<i}, \mathcal{A}(x) \ne \perp\right] \le \frac{12\epsilon}{k}, \tag{6}$$

otherwise the product in the product in the previous inequality would be less than $(1 - 12\epsilon/k)^{k/3} \le e^{-4\epsilon} < 1 - 2\epsilon$, contradicting the lower bound on this product.

For each $i \le k$, let $q_i(x)$ denote the number of queries that $\mathcal{A}$ makes to $x^{(i)}$ on input x. The query complexity of $\mathcal{A}$ guarantees that for each input x, $\sum_{i\le k} q_i(x) \le q$. Therefore, $\sum_{i\le k} \mathrm{E}_{x\sim\mu^k}[q_i(x)] \le q$ and at least $\frac{2}{3}k$ indices $i \in [k]$ satisfy

$$\mathrm{E}_{x\sim\mu^k}[q_i(x)] \le \frac{3q}{k}. \tag{7}$$

Thus, some index $i^* \in [k]$ satisfies both (6) and (7). Fix such an index i^*. For inputs $y \in \mu^k$ and $x \in \mu$, write $y^{(i^* \leftarrow x)} := (y^{(1)}, \ldots, y^{(i^*-1)}, x, y^{(i^*+1)}, \ldots, y^{(k)})$ to be the input obtained by replacing $y^{(i^*)}$ with x in y. With this notation, the two conditions (6) and (7) satisfied by i^* can be rewritten as

$$\mathop{\mathrm{E}}_{y\sim\mu^k}\left[\Pr_{x\sim\mu}\left[\mathcal{A}(y^{(i^*\leftarrow x)})_{i^*} \neq f(x) \;\middle|\; \mathcal{A}(y^{(i^*\leftarrow x)})_{<i^*} = f^k(y^{(i^*\leftarrow x)})_{<i^*}, \mathcal{A}(y^{(i^*\leftarrow x)}) \neq \perp\right]\right] \leq \frac{12\epsilon}{k}$$

and

$$\mathop{\mathrm{E}}_{y\sim\mu^k}\left[\mathop{\mathrm{E}}_{x\sim\mu}\left[q_{i^*}(y^{(i^*\leftarrow x)})\right]\right] \leq \frac{3q}{k}.$$

The correctness of $\mathcal{A}$ also guarantees that

$$\mathop{\mathrm{E}}_{y\sim\mu^k}\left[\Pr_{x\sim\mu}\left[\mathcal{A}(y^{(i^*\leftarrow x)}) = \perp\right]\right] \leq \delta$$

and

$$\mathop{\mathrm{E}}_{y\sim\mu^k}\left[\Pr_{x\sim\mu}\left[\mathcal{A}(y^{(i^*\leftarrow x)})_{<i^*} \neq f^k(y^{(i^*\leftarrow x)})_{<i^*}) \;\middle|\; \mathcal{A}(y^{(i^*\leftarrow x)}) \neq \perp\right]\right] \leq \epsilon.$$

Therefore, by Markov's inequality, there exists an input $z \in \{0,1\}^{n\times k}$ such that

$$\begin{aligned}
\Pr_{x\sim\mu}\left[\mathcal{A}(z^{(i^*\leftarrow x)}) = \perp\right] &\leq 4\delta,\\
\Pr_{x\sim\mu}\left[\mathcal{A}(z^{(i^*\leftarrow x)})_{<i^*} \neq f^k(z^{(i^*\leftarrow x)})_{<i^*} \;\middle|\; \mathcal{A}(z^{(i^*\leftarrow x)}) \neq \perp\right] &\leq 4\epsilon,\\
\Pr_{x\sim\mu}\left[\mathcal{A}(z^{(i^*\leftarrow x)})_{i^*} \neq f(x) \;\middle|\; \mathcal{A}(z^{(i^*\leftarrow x)})_{<i^*} = f^k(z^{(i^*\leftarrow x)}), \mathcal{A}(z^{(i^*\leftarrow x)}) \neq \perp\right] &\leq \frac{48\epsilon}{k}, \text{ and}\\
\mathop{\mathrm{E}}_{x\sim\mu}\left[q_{i^*}(z^{(i^*\leftarrow x)})\right] &\leq \frac{12q}{k}.
\end{aligned}$$

Let $\mathcal{A}'$ be the deterministic algorithm that computes $f(x)$ by simulating $\mathcal{A}$ on the input $z^{(i^*\leftarrow x)}$ with two additions:

1. If $\mathcal{A}$ attempts to query more than $\frac{120q}{k}$ bits of x, $\mathcal{A}'$ aborts, and
2. When $\mathcal{A}$ terminates, the algorithm $\mathcal{A}'$ first verifies that the output generated by $\mathcal{A}$ satisfies $\mathcal{A}(z^{(i^*\leftarrow x)})_{\leq i^*} = f^k(z^{(i^*\leftarrow x)})$. If so $\mathcal{A}'$ returns the value $\mathcal{A}(z^{(i^*\leftarrow x)})_{i^*}$; if not, $\mathcal{A}'$ aborts.

The algorithm $\mathcal{A}'$ has query complexity at most $\frac{120q}{k}$ and, by the conditions satisfied by z, it aborts with probability at most $\frac{1}{10} + 4\delta + 4\epsilon$ and errs with probability at most $\frac{48\epsilon}{k}$ when $x \sim \mu$. ◀

3.3 Proof of Theorem 2

We now complete the proof of Theorem 2. Fix $\delta = \frac{1}{40}$. By Proposition 14 and the second inequality of Lemma 15,

$$\overline{\mathrm{R}}_{\frac{96\epsilon}{k}}(f) \leq 2\,\mathrm{R}_{\frac{1}{2},\frac{48\epsilon}{k}}(f) \leq 2\,\mathrm{R}_{\frac{1}{5}+4\delta+4\epsilon,\frac{48\epsilon}{k}}(f) \leq 2\max_{\mu}\mathrm{D}^{\mu}_{\frac{1}{10}+2\delta+2\epsilon,\frac{24\epsilon}{k}}(f).$$

Let μ^* denote a distribution where the maximum is attained. By Lemma 16,

$$\mathrm{D}^{\mu^*}_{\frac{1}{10}+2\delta+2\epsilon,\frac{24\epsilon}{k}}(f) = O\left(\frac{1}{k}\cdot \mathrm{D}^{(\mu^*)^k}_{\frac{\delta}{2},\frac{\epsilon}{2}}(f^k)\right).$$

Using the first inequality of Lemma 15 we then obtain

$$\mathrm{D}^{(\mu^*)^k}_{\frac{\delta}{2},\frac{\epsilon}{2}}(f^k) \le \max_{\nu} \mathrm{D}^{\nu}_{\frac{\delta}{2},\frac{\epsilon}{2}}(f^k) \le \mathrm{R}_{\delta,\epsilon}(f^k).$$

Combining these inequalities and applying Proposition 14 once more yields

$$\overline{\mathrm{R}}_{\frac{96\epsilon}{k}}(f) \le O\big(\tfrac{1}{k} \cdot \mathrm{R}_{\delta,\epsilon}(f^k)\big) \le O\big(\tfrac{1}{k} \cdot \overline{\mathrm{R}}_{\epsilon}(f^k)\big).$$

Theorem 2 follows from the identity $\overline{\mathrm{R}}_{\frac{\epsilon}{k}}(f) = \Theta\big(\overline{\mathrm{R}}_{\frac{96\epsilon}{k}}(f)\big)$ obtained from the standard success amplification trick. ◀

3.4 Proof of Corollaries 3 and 4

Corollary 3 is obtained as a direct consequence of Theorems 1 and 2.

Proof of Corollary 3. The upper bound is via the universal bound (3). For the matching lower bound, let $f : \{0,1\}^n \to \{0,1\}$ be a function that satisfies the condition of Theorem 1. By Theorem 2, the randomized communication complexity of f^k satisfies

$$R(f^k) \ge \overline{\mathrm{R}}(f^k) = \Omega\big(k \cdot \overline{\mathrm{R}}_{\frac{1}{3k}}(f)\big)$$

By Theorem 1,

$$\overline{\mathrm{R}}_{\frac{1}{3k}}(f) = \Omega\big(\mathrm{R}(f) \cdot \log k\big)$$

as long as $k \le 2^{(\frac{n}{\log n})^{1/3}}$. Combining those inequalities yields $\mathrm{R}(f^k) = \Omega\big(k \log k \cdot \mathrm{R}(f)\big)$, as we wanted to show. ◀

The proof of Corollary 4 uses the following randomized query-to-communication lifting theorem of Göös, Pitassi, and Watson [17].

▶ **Theorem 17** (Göös, Pitassi, Watson). *Define* $\textsc{Ind}_m : [m] \times \{0,1\}^m \to \{0,1\}$ *to be the index function mapping* (x, y) *to* y_x *and fix* $m = n^{256}$*. For every* $f : \{0,1\}^n \to \{0,1\}$*,*

$$\mathrm{R}^{\mathrm{cc}}(f \circ \textsc{Ind}_m) = \mathrm{R}(f) \cdot \Theta(\log n)$$

and

$$\mathrm{R}^{\mathrm{cc}}(f^k \circ \textsc{Ind}_m) = \mathrm{R}(f^k) \cdot \Theta(\log n).$$

▶ Remark 18. The statement of Theorem 17 in [17] only mentions the first identity explicitly. However, as discussed in their Section II, the theorem statement holds for functions with any finite range.[2] Therefore, the theorem holds for the function f^k as well as f.

Proof of Corollary 4. By Corollary 3, there exists a function $f : \{0,1\}^n \to \{0,1\}$ which satisfies $\mathrm{R}(f^k) = \Theta(k \log k \cdot \mathrm{R}(f))$. Combining this result with Theorem 17, we obtain

$$\begin{aligned}
\mathrm{R}^{\mathrm{cc}}\big((f \circ \textsc{Ind}_m)^k\big) &= \mathrm{R}^{\mathrm{cc}}\big(f^k \circ \textsc{Ind}_m\big) \\
&= \mathrm{R}(f^k) \cdot \Theta(\log n) \\
&= \Theta(k \log k \cdot \mathrm{R}(f) \cdot \log n) \\
&= \Theta(k \log k) \cdot \mathrm{R}^{\mathrm{cc}}(f \circ \textsc{Ind}_m).
\end{aligned}$$ ◀

[2] In fact, their theorem also holds in even more general settings such as when f is a partial function or a relation, for example.

References

1. Scott Aaronson, Shalev Ben-David, and Robin Kothari. Separations in query complexity using cheat sheets. In *Proceedings 48th Annual ACM Symposium on Theory of Computing*, pages 863–876, 2016.
2. Andris Ambainis, Kaspars Balodis, Aleksandrs Belovs, Troy Lee, Miklos Santha, and Juris Smotrovs. Separations in query complexity based on pointer functions. *Journal of the ACM*, 64(5):32, 2017.
3. Andris Ambainis, Martins Kokainis, and Robin Kothari. Nearly optimal separations between communication (or query) complexity and partitions. In *Proceedings 31st Annual Conference on Computational Complexity*, page 4, 2016.
4. Anurag Anshu, Aleksandrs Belovs, Shalev Ben-David, Mika Göös, Rahul Jain, Robin Kothari, Troy Lee, and Miklos Santha. Separations in communication complexity using cheat sheets and information complexity. In *Proceedings 57th Annual IEEE Symposium on Foundations of Computer Science*, pages 555–564, 2016.
5. Ziv Bar-Yossef, Thathachar S Jayram, Ravi Kumar, and D Sivakumar. An information statistics approach to data stream and communication complexity. *Journal of Computer and System Sciences*, 68(4):702–732, 2004.
6. Boaz Barak, Mark Braverman, Xi Chen, and Anup Rao. How to Compress Interactive Communication. *SIAM Journal on Computing*, 42(3):1327–1363, 2013. doi:10.1137/100811969.
7. Yosi Ben-Asher and Ilan Newman. Decision Trees with AND, OR Queries. In *Proceedings 10th Annual Structure in Complexity Theory Conference*, pages 74–81, 1995. doi:10.1109/SCT.1995.514729.
8. Shalev Ben-David and Robin Kothari. Randomized Query Complexity of Sabotaged and Composed Functions. *Theory of Computing*, 14(1):1–27, 2018. doi:10.4086/toc.2018.v014a005.
9. Mark Braverman and Anup Rao. Information Equals Amortized Communication. *IEEE Transactions on Information Theory*, 60(10):6058–6069, 2014.
10. Harry Buhrman, Ilan Newman, Hein Röhrig, and Ronald de Wolf. Robust Polynomials and Quantum Algorithms. *Theory Comput. Syst.*, 40(4):379–395, 2007. doi:10.1007/s00224-006-1313-z.
11. Amit Chakrabarti, Yaoyun Shi, Anthony Wirth, and Andrew Chi-Chih Yao. Informational Complexity and the Direct Sum Problem for Simultaneous Message Complexity. In *Proceedings 42nd Annual IEEE Symposium on Foundations of Computer Science*, pages 270–278, 2001. doi:10.1109/SFCS.2001.959901.
12. Benny Chor, Oded Goldreich, Johan Håstad, Joel Friedman, Steven Rudich, and Roman Smolensky. The Bit Extraction Problem or t-Resilient Functions. In *Proceedings 26th Annual IEEE Symposium on Foundations of Computer Science*, pages 396–407, 1985. doi:10.1109/SFCS.1985.55.
13. Andrew Drucker. Improved direct product theorems for randomized query complexity. *Computational Complexity*, 21(2):197–244, 2012.
14. Tomás Feder, Eyal Kushilevitz, Moni Naor, and Noam Nisan. Amortized Communication Complexity. *SIAM Journal on Computing*, 24(4):736–750, 1995. doi:10.1137/S0097539792235864.
15. Anat Ganor, Gillat Kol, and Ran Raz. Exponential separation of information and communication. In *Proceedings 55th Annual IEEE Symposium on Foundations of Computer Science*, pages 176–185, 2014.
16. Mika Göös, Toniann Pitassi, and Thomas Watson. Deterministic communication vs. partition number. In *Proceedings 56th Annual IEEE Symposium on Foundations of Computer Science*, pages 1077–1088, 2015.
17. Mika Göös, Toniann Pitassi, and Thomas Watson. Query-to-Communication Lifting for BPP. In *Proceedings 58th Annual IEEE Symposium on Foundations of Computer Science*, 2017.
18. Russell Impagliazzo, Ran Raz, and Avi Wigderson. A Direct Product Theorem. In *Proceedings 9th Annual Structure in Complexity Theory Conference*, pages 88–96, 1994. doi:10.1109/SCT.1994.315814.

19 Rahul Jain, Hartmut Klauck, and Miklos Santha. Optimal direct sum results for deterministic and randomized decision tree complexity. *Inf. Process. Lett.*, 110(20):893–897, 2010. doi:10.1016/j.ipl.2010.07.020.

20 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.

21 Marco Molinaro, David P. Woodruff, and Grigory Yaroslavtsev. Beating the Direct Sum Theorem in Communication Complexity with Implications for Sketching. In *Proceedings 24th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1738–1756, 2013. doi:10.1137/1.9781611973105.125.

22 Marco Molinaro, David P Woodruff, and Grigory Yaroslavtsev. Amplification of one-way information complexity via codes and noise sensitivity. In *Proceedings 42nd Annual International Colloquium on Automata, Languages, and Programming*, pages 960–972. Springer, 2015.

23 Sagnik Mukhopadhyay and Swagato Sanyal. Towards Better Separation between Deterministic and Randomized Query Complexity. In *Proceedings 35th Annual Foundations of Software Technology and Theoretical Computer Science*, pages 206–220, 2015.

24 Noam Nisan, Steven Rudich, and Michael E. Saks. Products and Help Bits in Decision Trees. *SIAM Journal on Computing*, 28(3):1035–1050, 1999. doi:10.1137/S0097539795282444.

25 Ronen Shaltiel. Towards proving strong direct product theorems. *Computational Complexity*, 12(1-2):1–22, 2003.

26 Alexander Sherstov. The Power of Asymmetry in Constant-Depth Circuits. *SIAM Journal on Computing*, 47(6):2362–2434, 2018. doi:10.1137/16M1064477.

Relations and Equivalences Between Circuit Lower Bounds and Karp-Lipton Theorems

Lijie Chen
MIT, Cambridge, MA, USA

Dylan M. McKay
MIT, Cambridge, MA, USA

Cody D. Murray
No Affiliation

R. Ryan Williams
MIT, Cambridge, MA, USA

Abstract

A frontier open problem in circuit complexity is to prove $\mathsf{P}^{\mathsf{NP}} \not\subset \mathsf{SIZE}[n^k]$ for all k; this is a necessary intermediate step towards $\mathsf{NP} \not\subset \mathsf{P}_{/\mathsf{poly}}$. Previously, for several classes containing P^{NP}, including $\mathsf{NP}^{\mathsf{NP}}$, $\mathsf{ZPP}^{\mathsf{NP}}$, and $\mathsf{S}_2\mathsf{P}$, such lower bounds have been proved via *Karp-Lipton-style Theorems*: to prove $\mathcal{C} \not\subset \mathsf{SIZE}[n^k]$ for all k, we show that $\mathcal{C} \subset \mathsf{P}_{/\mathsf{poly}}$ implies a "collapse" $\mathcal{D} = \mathcal{C}$ for some larger class $\mathcal{D}$, where we already know $\mathcal{D} \not\subset \mathsf{SIZE}[n^k]$ for all k.

It seems obvious that one could take a different approach to prove circuit lower bounds for P^{NP} that does not require proving any Karp-Lipton-style theorems along the way. We show this intuition is *wrong*: **(weak) Karp-Lipton-style theorems for P^{NP} are *equivalent* to fixed-polynomial size circuit lower bounds for P^{NP}**. That is, $\mathsf{P}^{\mathsf{NP}} \not\subset \mathsf{SIZE}[n^k]$ for all k if and only if ($\mathsf{NP} \subset \mathsf{P}_{/\mathsf{poly}}$ implies $\mathsf{PH} \subset$ i.o.-$\mathsf{P}^{\mathsf{NP}}_{/n}$).

Next, we present new consequences of the assumption $\mathsf{NP} \subset \mathsf{P}_{/\mathsf{poly}}$, towards proving similar results for NP circuit lower bounds. We show that under the assumption, fixed-polynomial circuit lower bounds for NP, nondeterministic polynomial-time derandomizations, and various fixed-polynomial time simulations of NP are all *equivalent*. Applying this equivalence, we show that **circuit lower bounds for NP imply better Karp-Lipton collapses**. That is, if $\mathsf{NP} \not\subset \mathsf{SIZE}[n^k]$ for all k, then for all $\mathcal{C} \in \{\oplus\mathsf{P}, \mathsf{PP}, \mathsf{PSPACE}, \mathsf{EXP}\}$, $\mathcal{C} \subset \mathsf{P}_{/\mathsf{poly}}$ implies $\mathcal{C} \subset$ i.o.-$\mathsf{NP}_{/n^\varepsilon}$ for all $\varepsilon > 0$. Note that unconditionally, the collapses are only to MA and not NP.

We also explore consequences of circuit lower bounds for a *sparse* language in NP. Among other results, we show if a polynomially-sparse NP language does not have $n^{1+\varepsilon}$-size circuits, then $\mathsf{MA} \subset$ i.o.-$\mathsf{NP}_{/O(\log n)}$, $\mathsf{MA} \subset$ i.o.-$\mathsf{P}^{\mathsf{NP}[O(\log n)]}$, and $\mathsf{NEXP} \not\subset \mathsf{SIZE}[2^{o(m)}]$. Finally, we observe connections between these results and the "hardness magnification" phenomena described in recent works.

2012 ACM Subject Classification Theory of computation → Circuit complexity

Keywords and phrases Karp-Lipton Theorems, Circuit Lower Bounds, Derandomization, Hardness Magnification

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.30

Funding Supported by NSF CCF-1741615 (Common Links in Algorithms and Complexity).

Acknowledgements Part of this work was completed while three of the authors were visiting the Simons Institute at UC Berkeley, as part of the program on Lower Bounds in Computational Complexity. We thank them for their hospitality and excellent environment. We also thank Josh Alman for helpful last-minute proofreading, and the CCC reviewers for useful comments.

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka; Article No. 30; pp. 30:1–30:21

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

1 Introduction

Let $\mathcal{C}$ be a complexity class containing NP. A longstanding method for proving fixed-polynomial circuit lower bounds for functions in $\mathcal{C}$, first observed by Kannan [25], applies versions of the classical Karp-Lipton Theorem in a particular way:

1. If $\mathsf{NP} \not\subset \mathsf{P}_{/\mathsf{poly}}$, then $\mathsf{SAT} \in \mathsf{NP} \subset \mathcal{C}$ does not have polynomial-size circuits.
2. If $\mathsf{NP} \subset \mathsf{P}_{/\mathsf{poly}}$, then by a "collapse" theorem, we have $\mathsf{PH} \subseteq \mathcal{C}$. But for every k, there is an $f \in \mathsf{PH}$ that does not have n^k-size circuits, so we are also done.

Such collapse theorems are called *Karp-Lipton Theorems*, as they were first discovered by Karp and Lipton [26] in their pioneering work on complexity classes with advice. The general theme of such theorems is a connection between non-uniform and uniform complexity:

> "$\mathcal{C}$ has (non-uniform) polynomial-size circuits implies a collapse of (uniform) complexity classes."

Over the years, Karp-Lipton Theorems have been applied to prove circuit lower bounds for the complexity classes $\mathsf{NP}^{\mathsf{NP}}$ [25], $\mathsf{ZPP}^{\mathsf{NP}}$ [6, 27], $\mathsf{S}_2\mathsf{P}$ [9, 10], PP [37, 1][1], and Promise-MA and MA/1 [33].[2] Other literature on Karp-Lipton Theorems include [38, 12, 13].

When one first encounters such a lower bound argument, the non-constructivity of the result (the two uncertain cases) and the use of a Karp-Lipton Theorem looks strange.[3] It appears obvious that one ought to be able to prove circuit lower bounds in a fundamentally *different way*, without worrying over any collapses of the polynomial hierarchy. It is easy to imagine the possibility of a sophisticated combinatorial argument establishing a lower bound for P^{NP} functions (one natural next step in such lower bounds) which has nothing to do with simulating PH more efficiently, and has no implications for it.

P^{NP} Circuit Lower Bounds are Equivalent to Karp-Lipton Collapses to P^{NP}. We show that, in a sense, the above intuition is **false**: any fixed-polynomial-size circuit lower bound for P^{NP} would imply a Karp-Lipton Theorem collapsing PH all the way to P^{NP}. (There are some technicalities: the P^{NP} simulation uses small advice and only works infinitely often, but we believe these conditions can potentially be removed, and they do not change the moral of our story.) We find this result surprising; it shows that *in order to prove a circuit lower bound for P^{NP}, one cannot avoid proving a Karp-Lipton Theorem for P^{NP} in the process.* A Karp-Lipton Theorem is both necessary and sufficient for such lower bounds.

▶ **Theorem 1** (P^{NP} Circuit Lower Bounds are Equivalent to a Karp-Lipton Collapse to P^{NP}). *$P^{NP} \not\subset SIZE[n^k]$ for all k if and only if ($NP \subset P_{/\mathsf{poly}} \implies PH \subset$ i.o.-$P^{NP}_{/n}$).*

One direction of Theorem 1 follows immediately from the classical lower bound paradigm described above. In particular, assuming $\mathsf{P}^{\mathsf{NP}} \subset \mathsf{SIZE}[n^k]$ for some k and assuming $\mathsf{NP} \subset \mathsf{P}_{/\mathsf{poly}} \implies \mathsf{PH} \subset$ i.o.-$\mathsf{P}^{\mathsf{NP}}_{/n}$ we have

$$\mathsf{PH} \subset \text{i.o.-}\mathsf{P}^{\mathsf{NP}}_{/n} \subseteq \text{i.o.-}\mathsf{SIZE}[O(n)^k],$$

[1] Both Vinodchandran and Aaronson's proofs of $\mathsf{PP} \not\subset \mathsf{SIZE}[n^k]$ use the Karp-Lipton-style theorem "$\mathsf{PP} \subset \mathsf{P}/\mathrm{poly}$ then $\mathsf{PP} = \mathsf{MA}$", which follows from [28]. Aaronson shows further that "$\mathsf{PP} \subset \mathsf{P}/\mathrm{poly}$ then $\mathsf{P}^{\mathsf{PP}} = \mathsf{MA}$". From there, one can directly construct a function in P^{PP} without n^k-size circuits.

[2] Santhanam used the Karp-Lipton-style theorem "$\mathsf{PSPACE} \subset \mathsf{P}_{/\mathsf{poly}}$ implies $\mathsf{PSPACE} = \mathsf{MA}$" to prove lower bounds against Promise-MA and MA with one bit of advice.

[3] Note Cai and Watanabe [11] found a constructive proof for $\mathsf{NP}^{\mathsf{NP}}$.

which contradicts known fixed-polynomial lower bounds for PH. The interesting direction is the converse, showing that *proving lower bounds against* P^{NP} ***implies*** *proving a Karp-Lipton collapse to* P^{NP} *that is sufficient for the lower bound.*

NP Circuit Lower Bounds Imply Better Karp-Lipton Collapses. After observing Theorem 1, a natural question is whether such a theorem holds for NP circuit lower bounds as well:

Does $NP \not\subset SIZE[n^k]$ *for all* k *imply a Karp-Lipton Collapse to NP?*

While we have not yet been able to prove this under the hypothesis $\mathsf{NP} \subset \mathsf{P/poly}$ as above, we can show it for stronger hypotheses. Another class of Karp-Lipton Theorems (used in circuit lower bounds for PP [37, 1] and Promise-MA [33]) give stronger collapses under hypotheses like $\mathsf{PSPACE} \subset \mathsf{P/poly}$: for any class $\mathcal{C}$ which is one of NEXP [24], $\mathsf{EXP}^{\mathsf{NP}}$ ([7] and [5]), EXP and PSPACE [5], PP [28] and $\oplus$P [23], we have:

If $\mathcal{C} \subset \mathsf{P/poly}$ then $\mathcal{C} \subseteq \mathsf{MA}$.

We show how NP circuit lower bounds can be used to derandomize MA. In fact, under the hypothesis $\mathsf{NP} \subset \mathsf{P/poly}$, we prove an equivalence between NP circuit lower bounds, fast Arthur-Merlin simulations of NP, and nondeterministic derandomization of Arthur-Merlin protocols.

To state our results, we first define a variation of the "robust simulation" which was originally introduced in [17]. For a complexity class $\mathcal{C}$ and a language L, we say L is in c-r.o.-$\mathcal{C}$ for a constant c, if there is a language $L' \in \mathcal{C}$ such that there are infinitely many m's such that for all $n \in [m, m^c]$, L' agrees with L on inputs of length n.[4] (See Section 2.1 for formal definitions.)

▶ **Theorem 2.** *Assuming* $NP \subset P_{/\mathsf{poly}}$*, the following are equivalent:*

1. NP *is not in* $SIZE[n^k]$ *for all* k*.*
2. $AM_{/1}$ *is in c-r.o.-*$NP_{/n^\varepsilon}$ *for all* $\varepsilon > 0$ *and integers* c*.*
   That is, Arthur-Merlin games with $O(1)$ *rounds and small advice can be simulated "c-robustly often" in NP with modest advice, for all constants* c*.*[5]
3. NP *does not have* n^k*-size witnesses for all* k*.*
   That is, for all k*, there is a language* $L \in NP$*, a poly-time verifier* V *for* L*, and infinitely many* $x_n \in L$ *such that* $V(x_n, \cdot)$ *has no witness of circuit complexity at most* n^k*.*
4. *For all* k *and* d*, there is a polynomial-time nondeterministic PRG with seed-length* $O(\log n)$ *and* n *bits of advice against* n^k*-size circuits* d*-robustly often.*[6]
5. NP *is not in* $AMTIME(n^k)$ *for all* k*.*
6. $(NP \cap coNP)_{/n^\varepsilon}$ *is not in* $SIZE[n^k]$ *for all* k *and all* $\varepsilon > 0$*.*
7. $(AM \cap coAM)_{/1}$ *is in c-r.o.-*$(NP \cap coNP)_{/n^\varepsilon}$ *for all* $\varepsilon > 0$ *and all integers* c*.*

That is, under $\mathsf{NP} \subset \mathsf{P/poly}$, the tasks of fixed-polynomial lower bounds for NP, lower bounds for $(\mathsf{NP} \cap \mathsf{coNP})/n^\varepsilon$, uniform lower bounds on simulating NP within AM, and derandomizing AM in NP are all equivalent.

[4] The original definition of $L \subseteq$ r.o.-$\mathcal{C}$ requires that there is a *single* language $L' \in \mathcal{C}$ such that for all c there are infinitely many m's such that for all $n \in [m, m^c]$, L' agrees with L on inputs of length n.

[5] See the Preliminaries for a definition of "c-robustly often". Intutively, it is a mild strengthening of "infinitely often".

[6] See the Preliminaries for formal definitions.

We recall another type of Karp-Lipton collapse was shown by [4]: $\mathsf{NP} \subset \mathsf{P}_{/\mathsf{poly}}$ implies $\mathsf{AM} = \mathsf{MA}$. An intriguing corollary of Theorem 2 is that fixed-polynomial lower bounds for NP would improve this collapse, from MA to r.o.-c-$\mathsf{NP}_{/n^\varepsilon}$ for all c:

▶ **Corollary 3** (NP Circuit Lower Bounds Equivalent to a Karp-Lipton Collapse of AM to NP)**.** *$\mathsf{NP} \not\subset \mathsf{SIZE}[n^k]$ for all k if and only if ($\mathsf{NP} \subset \mathsf{P}_{/\mathsf{poly}} \implies$ AM is in r.o.-c-$\mathsf{NP}_{/n^\varepsilon}$ for all c).*

Another consequence of Theorem 2 is that NP circuit lower bounds imply better Karp-Lipton collapses from MA down to NP:

▶ **Theorem 4** (NP Circuit Lower Bounds Imply Better Karp-Lipton Collapses)**.** *Let $\mathcal{C} \in \{\oplus\mathsf{P}, \mathsf{PSPACE}, \mathsf{PP}, \mathsf{EXP}\}$. Suppose $\mathsf{NP} \not\subset \mathsf{SIZE}[n^k]$ for all k. Then for all $\varepsilon > 0$, ($\mathcal{C} \subset \mathsf{P}_{/\mathsf{poly}} \implies \mathcal{C} \subset$ i.o.-$\mathsf{NP}_{/n^\varepsilon}$). In particular, polynomial-size circuits for any $\mathcal{C}$-complete language L can be constructed in NP infinitely often, with n^ε advice.*

▶ Remark 5. By "circuits for L can be constructed in NP infinitely often", we mean that there is a nondeterministic poly-time algorithm A such that, for infinitely many n, A on input 1^n outputs a circuit C_n for L_n on at least one computation path, and on all paths where such a C_n is not output, A outputs *reject*.

Consequences of Weak Circuit Lower Bounds for Sparse Languages in NP. Theorem 2 shows that assuming $\mathsf{NP} \subset \mathsf{P}_{/\mathsf{poly}}$, fixed-polynomial lower bounds for NP imply $\mathsf{AM} = \mathsf{MA} \subseteq$ i.o.-$\mathsf{NP}_{/n^\varepsilon}$. This is also the reason that we can only show collapses to i.o.-$\mathsf{NP}_{/n^\varepsilon}$ in Theorem 4. It is interesting to ask whether the n^ε advice in the simulation can be eliminated or reduced. In the following, we show that an $n^{1.00001}$-size circuit lower bound for a polynomially-sparse language in NP would imply an advice reduction, along with other interesting consequences.

▶ **Theorem 6** (Consequences of Weak Circuit Lower Bounds for Polynomially-Sparse NP Languages)**.** *Suppose there is an $\varepsilon > 0$, a $c \geq 1$, and an n^c-sparse $L \in \mathsf{NP}$ without $n^{1+\varepsilon}$-size circuits. Then $\mathsf{MA} \subset$ i.o.-$\mathsf{NP}_{/O(\log n)}$, $\mathsf{MA} \subseteq$ i.o.-$\mathsf{P}^{\mathsf{NP}[O(\log n)]}$, and $\mathsf{NE} \not\subset \mathsf{SIZE}[2^{\delta \cdot n}]$ for some $\delta > 0$ (which implies $\mathsf{NP} \not\subset \mathsf{SIZE}[n^k]$ for all k).*

One step in the proof of Theorem 6 is a form of *hardness condensation* (as termed by Impagliazzo [21]) for sparse NP languages. The goal of hardness condensation [8, 22] is that, given a function f on n input bits with complexity S, we want to construct a function $\widetilde{f}$ on $\ell \ll n$ input bits that still has complexity roughly S. We show how a hard $S(n)$-sparse language in $\mathsf{NTIME}[T(n)]$ can be "condensed" in a generic way, based on the sparsity $S(n)$. We can efficiently build a PRG from the harder condensed function.

Theorem 6 shows how a very weak lower bound ($n^{1+\varepsilon}$) for a sparse language $L \in \mathsf{NP}$ would imply an *exponential-size* lower bound for NE (note, the converse is easy to show). This is reminiscent of a recent line of work [32, 31, 29] on "hardness magnification" phenomena, showing that seemingly weak circuit lower bounds for certain problems can in fact imply strong circuit lower bounds which are out of reach of current proof techniques.

At a high level, the hardness magnification results in the above-cited papers show how weak lower bounds on "compression problems" can imply strong complexity class separations. These compression problems have the form: *given a string, does it have a small efficient representation?* As an example, in the Minimum Circuit Size Problem for size $S(m) \ll 2^m$, denoted as $\mathsf{MCSP}[S(m)]$, we are given a truth table of length $N = 2^m$ and want to know if the function has a circuit of size at most $S(m)$. As an example of hardness magnification, McKay, Murray, and Williams [29] show that, if there is an $\varepsilon > 0$ such that $\mathsf{MCSP}[2^{m/\log^\star m}]$ is not in $\mathsf{SIZE}[N^{1+\varepsilon}]$, then $\mathsf{NP} \not\subset \mathsf{P}_{/\mathsf{poly}}$. Thus a very weak circuit size lower bound for $\mathsf{MCSP}[2^{m/\log^\star m}]$ would imply a super-polynomial lower bound for SAT!

Sparsity Alone Implies a Weak Hardness Magnification. We identify a simple property of all efficient compression problems which alone implies a (weak) form of hardness magnification: the *sparsity of the underlying language*. For any compression problem on length-N strings where we ask for a length-$\ell(N)$ representation (think of $\ell(N) \leq n^{o(1)}$), there are at most $2^{\ell(N)}$ strings in the language. Scaling up the sparsity of Theorem 6, we show that non-trivial circuit lower bounds for *any* NP problem with subexponential sparsity already implies longstanding circuit lower bounds. In fact, we have an equivalence:

▶ **Theorem 7.** *NEXP $\not\subset$ $\mathsf{P}_{/\mathsf{poly}}$ if and only if there exists an $\varepsilon > 0$ such that for every sufficiently small $\beta > 0$, there is a 2^{n^β}-sparse language $L \in \mathit{NTIME}[2^{n^\beta}]$ without $n^{1+\varepsilon}$-size circuits.*

It follows that an $n^{1+\varepsilon}$-size circuit lower bound for $\mathsf{MCSP}[2^{m/\log^\star m}]$ implies $\mathsf{NEXP} \not\subset \mathsf{P}_{/\mathsf{poly}}$. We remark while the lower bound consequence here is *much weaker* than the consequences of prior work [32, 31, 29] (only $\mathsf{NEXP} \not\subset \mathsf{P}_{/\mathsf{poly}}$, instead of $\mathsf{NP} \not\subset \mathsf{P}_{/\mathsf{poly}}$), the hypothesis has much more flexibility: Theorem 7 allows for any sparse language in $\mathsf{NTIME}[2^{n^{o(1)}}]$, while the MCSP problem is in $\mathsf{NTIME}[n^{1+o(1)}]$.[7]

Finally, we observe that Theorem 7 is similar in spirit to the Hartmanis-Immerman-Sewelson theorem [20] which states that there is a polynomially-sparse language in $\mathsf{NP} \setminus \mathsf{P}$ if and only if $\mathsf{NE} \neq \mathsf{E}$. Theorem 7 can be interpreted as a certain optimized, non-uniform analogue of Hartmanis-Immerman-Sewelson theorem, in a different regime of sparsity.

Organization of the Paper. In Section 2, we introduce the necessary preliminaries for this paper. In Section 3, we prove that fixed-polynomial circuit lower bounds for P^{NP} is equivalent to a (weak) Karp-Lipton theorem for P. In Section 4, we prove our equivalence theorem for NP circuit lower bounds, fast simulations of NP, and nondeterministic polynomial-time derandomization, under the hypothesis $\mathsf{NP} \subset \mathsf{P}/\mathrm{poly}$. In Section 5, we show how our equivalence theorem implies that fixed polynomial circuit lower bounds for NP implies better Karp-Lipton theorems for higher complexity classes. In Section 6, we prove the consequences of weak circuit lower bounds for sparse NP languages. Finally, in Section 7, we discuss some interesting open questions stemming from this work.

2 Preliminaries

We assume basic knowledge of complexity theory (see e.g. [3, 19] for excellent references). Here we review some notation and concepts that are of particular interest for this paper.

Notation. All languages considered are over $\{0,1\}$. For a language L, we define $L_n := \{0,1\}^n \cap L$. For $s : \mathbb{N} \to \mathbb{N}$, $\mathsf{SIZE}[s(n)]$ is the class of languages decided by an infinite circuit family where the nth circuit in the family has size at most $s(n)$. $\oplus\mathsf{P}$ is the closure under polynomial-time reductions of the decision problem Parity-SAT: *Given a Boolean formula, is the number of its satisfying assignments odd?*

For a deterministic or nondeterministic class $\mathcal{C}$ and function $a(n)$, $\mathcal{C}/a(n)$ is the class of languages L such that there is an $L' \in \mathcal{C}$ and function $f : \mathbb{N} \to \{0,1\}^\star$ with $|f(n)| \leq a(n)$ for all x, such that $L = \{x \mid (x, f(|x|)) \in L'\}$. That is, the advice string $f(n)$ can be used to solve all n-bit instances within class $\mathcal{C}$. For "promise" classes $\mathcal{C}$ such as MA and AM, $\mathcal{C}/a(n)$ is defined similarly, except that the promise of the class is only required to hold when the correct advice $f(n)$ is provided.

[7] We remark that these results are not directly related to hardness magnification for NC^1-complete problems [2, 15], as the problems studied in these works are clearly not sparse.

2.1 Infinitely Often and Robust Simulations

In this section, let $\mathscr{C}$ be a class of languages. Here we recall infinitely often and robust simulations, the latter of which was first defined and studied in [17]. Robust simulations expand on the notion of "infinitely often" simulations. A language $L \in$ i.o.-$\mathscr{C}$ (*infinitely often* $\mathscr{C}$), if there is a language L' in $\mathscr{C}$ such that there are infinitely many n such that $L_n = L'_n$. A language $L \in$ r.o.-$\mathscr{C}$ (*robustly often* $\mathscr{C}$), if there is a language L' in $\mathscr{C}$ such that for all $k \geq 1$, there are infinitely many n such that $L_m = L'_m$ for all $m \in [n, n^k]$. In this case, we say L' *r.o.-computes* L.

c-Robust Simulations. We consider a parameterized version of the robust simulation concept which is useful for stating our results. Let $c \geq 1$ be an integer constant. We say a language $L \in$ c-r.o.-$\mathscr{C}$ (c-robustly often $\mathscr{C}$) if there is an $L' \in \mathscr{C}$ and infinitely many n such that $L_m = L'_m$ for all $m \in [n, n^c]$. In this case, we say L' c-r.o.-computes L. Note that $L \in$ r.o.-$\mathscr{C}$ implies $L \in$ c-r.o.-$\mathscr{C}$ for all c, but the converse is not necessarily true.

More generally, a property $P(n)$ holds c-robustly often (c-r.o.-) if for all integers k, there are infinitely many m's such that $P(n)$ holds for all $n \in [m, m^c]$.

2.2 Non-deterministic Pseudo-Random Generators

Let $w(n), s(n) : \mathbb{N} \to \mathbb{N}$, and let $\mathscr{C}$ be a class of functions. We say a function family G, specified by $G_n : \{0,1\}^{w(n)} \times \{0,1\}^{s(n)} \to \{0,1\}^* \cap \{\perp\}$, is a *nondeterministic PRG against* $\mathscr{C}$ if for all sufficiently large n and all $C \in \mathscr{C}$, the following hold:

- For all $y \in \{0,1\}^{w(n)}$, either $G_n(y,z) \neq \perp$ for all z's (such a y is called *good*), or $G_n(y,z) = \perp$ for all z's (a *bad* y).
- There is at least one good $y \in \{0,1\}^{w(n)}$.
- Suppose $y \in \{0,1\}^{w(n)}$ is good, C has m input bits, and $|G_n(y,z)| \geq m$ for all z. Then

$$\left| \Pr_{z \in \{0,1\}^{s(n)}}[C(G_n(y,z)) = 1] - \Pr_{z \in \{0,1\}^m}[C(z) = 1] \right| < 1/n.$$

  As usual, if C takes less than $|G_n(y,z)|$ inputs, $C(G_n(y,z))$ corresponds to feeding C with the first m bits of $G_n(y,z)$.

Usually we are only interested in the seed length parameter $s(n)$ and the running time $T(n)$ of the PRG G_n as a function of n. To be concise, we say G is a $T(n)$-*time NPRG of seed length* $s(n)$ *against* $\mathscr{C}$.

We say G is a i.o.-NPRG or r.o.-NPRG, if it only fools functions in $\mathscr{C}$ infinite often or robustly often.

2.3 Circuit Complexity of Strings and Pseudorandom Generators

For a circuit C on ℓ inputs, we define the truth-table of C, denoted $tt(C) \in \{0,1\}^{2^\ell}$, to be the evaluation of C on all possible inputs sorted in lexicographical order. For every string y, let 2^ℓ be the smallest power of 2 such that $2^\ell > |y|$. We define the circuit complexity of y, denoted as $CC(y)$, to be the circuit complexity of the ℓ-input function defined by the truth-table $y10^{2^\ell - |y| - 1}$. We will use the following strong construction of pseudorandom generators from hard functions:

▶ **Theorem 8** (Umans [36]). *There is a constant g and a function $G : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ such that, for all s and Y satisfying $CC(Y) \geq s^g$, and for all circuits C of size s,*

$$\left| \Pr_{x \in \{0,1\}^{g \log |Y|}} [C(G(Y,x)) = 1] - \Pr_{x \in \{0,1\}^s} [C(x) = 1] \right| < 1/s.$$

Furthermore, G is computable in $\mathrm{poly}(|Y|)$ *time.*

Fortnow-Santhanam-Williams [18]. A work related to this paper is that of Fortnow, Santhanam, and Williams, who proved the equivalences $\mathsf{NP} \not\subset \mathsf{SIZE}[n^k]$ for all $k \iff \mathsf{P}^{\mathsf{NP}[n^k]} \not\subset \mathsf{SIZE}[n^c]$ for all k, c and $\mathsf{AM} \not\subset \mathsf{SIZE}[n^k]$ for all $k \iff \mathsf{MA} \not\subset \mathsf{SIZE}[n^k]$ for all k. We use intermediate results of theirs in our equivalence theorems (see the citations).

3 P^{NP} Circuit Lower Bounds Equivalent to Karp-Lipton Collapses to P^{NP}

In this section we prove Theorem 1 (restated below).

▶ **Reminder of Theorem 1.** *$P^{NP} \not\subset SIZE[n^k]$ for all k if and only if ($NP \subset P_{/\mathrm{poly}} \implies PH \subset$ i.o.-$P^{NP}_{/n}$).*

We begin with a lemma on the simulation of poly-time functions with an NP oracle. Essentially it says that if functions with an NP oracle always output strings of low circuit complexity, then we can simulate P^{NP} extremely efficiently in the polynomial hierarchy. This is similar in spirit to Fortnow, Santhanam, and Williams' result that $\mathsf{P}^{\mathsf{NP}} \subset \mathsf{SIZE}[n^k]$ implies $\mathsf{NP} \subseteq \mathsf{MATIME}(n^{O(k)})$ [18]; our result is more complex in that we simulate all of P^{NP}.

▶ **Lemma 9.** *Suppose there is a k such that for all FP^{NP} functions f, the circuit complexity of $f(x)$ is at most $|x|^k$ for all but finite many x. Then $P^{NP} \subseteq \Sigma_3 TIME[n^{O(k)}]$.*

Proof. Let $L \in \mathsf{P}^{\mathsf{NP}}$ be a language which can be computed by a 3-SAT oracle machine M in n^c time, for a constant c. Without loss of generality, we may assume M is a single-tape machine.

The $\mathsf{FP}^{\mathsf{NP}}$ Function f_{sol}. Consider the following $\mathsf{FP}^{\mathsf{NP}}$ function f_{sol}:

> $\mathsf{FP}^{\mathsf{NP}}$ function f_{sol} for printing assignments to all satisfiable oracle queries
>
> - Given an input x, simulate the 3-SAT oracle machine M running on the input x.
> - On the i-th step, if M makes an oracle query ψ (ψ is a 3-SAT instance) and ψ is satisfiable, call the NP oracle multiple times to construct a satisfying assignment for ψ, and print it. Letting m be the length of the assignment (note that $m \leq n^c$), we print $n^c + 1 - m$ additional ones.
> - Otherwise, print $n^c + 1$ zeros on the i-th step.

In the following we always assume n is sufficiently large. For all x with $|x| = n$, by assumption we know the string $f_{\mathsf{sol}}(x)$ has an n^k size circuit. Let ψ be a 3-SAT query made on i-th step which is satisfiable; ψ has a satisfying assignment corresponding to a sub-string of $f_{\mathsf{sol}}(x)$ starting from the position $(i-1) \cdot (n^c+1) + 1$, and therefore has circuit complexity at most $O(n^k) \leq n^{k+1}$. In particular, we can define a circuit $E_i(j) := f_{\mathsf{sol}}(x)((i-1) \cdot (n^c+1) + j)$ whose truth table encodes a SAT assignment to ψ.

The $\mathsf{FP}^{\mathsf{NP}}$ Function f_{history}. Next, we define a function $\mathsf{FP}^{\mathsf{NP}}$ function f_{history}, which prints the computation history of M. More precisely, we can interpret $f_{\mathsf{history}}(x)$ as a matrix $\mathsf{cell}(x) \in \Sigma^{n^c \times n^c}$, such that $\mathsf{cell}(i,j)$ represents the state of the j-th cell of the working tape before the i-th step, and Σ is a constant-size alphabet which represents all possible states of a cell. From our assumption, for an x with $|x| = n$, we know that $f_{\mathsf{history}}(x)$ has an n^k-size circuit.

The Algorithm. Now we are ready to describe a Σ_3 algorithm for L running in $n^{O(k)}$ time. At a high level, the algorithm first guesses two circuits C_{history} and C_{sol}, whose truth-tables are supposed to represent $f_{\mathsf{history}}(x)$ and $f_{\mathsf{sol}}(x)$, it tries to verify that these circuits correspond to a correct accepting computation of M on x. The whole verification can be done in $\Pi_2\mathsf{TIME}[n^{O(k)}]$, utilizing the fact that M is making 3-SAT queries. The formal description of the algorithm is given below.

A $\Sigma_3\mathsf{TIME}[n^{O(k)}]$ algorithm for L

(1) Given an input x, guess two n^k-size circuits C_{history} and C_{sol} where the truth-table of C_{history} is intended to be $f_{\mathsf{history}}(x)$), and the truth-table of C_{sol} is intended to be $f_{\mathsf{sol}}(x)$. Let $\mathsf{cell} \in \Sigma^{n^c \times n^c}$ be the matrix (tableau) corresponding to the truth-table of C_{history}.

(2) We check that C_{history} is consistent and accepting, assuming its claimed answers to oracle queries are correct. In particular, we universally check over all $(i,j) \in [n^c] \times [n^c]$ that $\mathsf{cell}(i,j)$ is consistent with the contents of $\mathsf{cell}(i-1,j-1), \mathsf{cell}(i-1,j), \mathsf{cell}(i,j+1)$ when $i > 1$, whether it agrees with the initial configuration when $i = 1$, and whether M is in an accept state when $i = n^c$.

(3) We check that the claimed answers to oracle queries in C_{history} are correct. For convenience, we assume the query string always starts at the leftmost position on the tape. We universally check over all step $i \in [n^c]$:

If there is no query at the i-th step, we *accept*.

(A) Let ψ be the 3-SAT query. If the claimed answer in C_{history} for ψ is *yes*, we examine the corresponding sub-string of $tt(C_{\mathsf{sol}})$, and check universally over all clauses in ψ that it is satisfied by the corresponding assignment in $tt(C_{\mathsf{sol}})$ (accepting if the check passes and rejecting if it fails).

(B) If the claimed answer in C_{history} for ψ is *no*, we universally check over all n^{k+1}-size circuits D that $tt(D)$ is not an assignment to ψ, by existentially checking that there is a clause in ψ which is not satisfied by $tt(D)$.

Running Time. It is straightforward to see that the above is a $\Sigma_3\mathsf{TIME}[n^{O(k)}]$ algorithm.

Correctness. When $x \in L$, there are C_{sol} and C_{history} such that $tt(C_{\mathsf{sol}})$ and $(\mathsf{C}_{\mathsf{history}})$ correspond to $f_{\mathsf{sol}}(x)$ and $f_{\mathsf{history}}(x)$, so all of the checks pass and the above algorithm accepts x.

Let $x \notin L$. We want to show that all possible n^k-size circuits for C_{history} and C_{sol} will be rejected. Assume for contradiction that there are circuits C_{history} and C_{sol} that can pass the whole verification. By our checks in step (2) of the algorithm, C_{history} is consistent and ends in accept state; therefore, at least one answer to its oracle queries is not correct. Suppose the

first incorrect answer occurs on the i-th step. Since C_{history} is consistent and all queries made before the i-th are correctly answered, the i-th query ψ is actually the correct i-th query made by machine M on the input x.

Therefore, if the correct answer to ψ is yes but C_{history} claims it is no, case (B) will not be passed, as there is always a satisfying assignment that can be represented by the truth-table of an n^{k+1}-size circuit. Similarly, if C_{history} incorrectly claims the answer is yes, then case (A) cannot be passed, as ψ is unsatisfiable. ◀

We are now ready to prove Theorem 1.

Proof of Theorem 1. Suppose (1) P^{NP} does not have $\mathsf{SIZE}[n^k]$ circuits for any fixed k and (2) $\mathsf{NP} \subset \mathsf{P}_{/\mathsf{poly}}$. By assumption (2), we have that for every c, $\Sigma_3\mathsf{TIME}[n^c] \subset \mathsf{SIZE}[n^{O(c)}]$. Therefore, applying (1), $\mathsf{P}^{\mathsf{NP}} \not\subseteq \Sigma_3\mathsf{TIME}[n^c]$ for every c. By the contrapositive of Lemma 9, for every k there is a P^{NP} function B that for infinitely many x of length n, the circuit complexity of $B(x)$ is greater than n^k. In other words, $B(x)$ outputs the truth tables of hard functions on infinitely many x.

Assumption (2) also implies a collapse of the polynomial hierarchy to $\mathsf{ZPP}^{\mathsf{NP}}$ [27]. By (2), we also have $\mathsf{ZPP}^{\mathsf{NP}} \subset \mathsf{P}_{/\mathsf{poly}}$, so every $\mathsf{ZPP}^{\mathsf{NP}}$ algorithm A has polynomial-size circuits, and thus by standard hardness-to-PRG constructions (e.g., Theorem 8) there is a fixed k such that a string of circuit complexity at least n^k can be used to construct a PRG that fools algorithm A on inputs of length n. As shown above, there is a function B in P^{NP} that can produce such strings on infinitely many inputs x. If the inputs x that make B produce high complexity strings are given as advice, then the $\mathsf{ZPP}^{\mathsf{NP}}$ algorithm A can be simulated in $\mathsf{P}^{\mathsf{NP}}_{/n}$: first, call B on the advice x to generate a hard function, produce a PRG of seed length $O(\log n)$ with the hard function, then simulate A on the input and the pseudorandom strings output by the PRG, using the NP oracle to simulate the NP oracle of A. Thus we have $\mathsf{ZPP}^{\mathsf{NP}} \subset \text{i.o.-}\mathsf{P}^{\mathsf{NP}}/n$.

Finally, we note that the n bits of advice can be reduced to n^ε bits for any desired $\varepsilon > 0$. For every $k > 0$, we can find an $\mathsf{FP}^{\mathsf{NP}}$ function that outputs a string of circuit complexity greater than n^k. Setting $k' = k/\varepsilon$, we can use an n^ε-length input as advice, and still get a function that is hard enough to derandomize ($(n^\varepsilon)^{k'} = (n^\varepsilon)^{k/\varepsilon} = n^k$). ◀

4 An Equivalence Theorem Under $\mathsf{NP} \subset \mathsf{P}_{/\mathsf{poly}}$

In this section we prove Theorem 2 together with several applications.

First, we need a strong size lower bound for a language in $(\mathsf{MA} \cap \mathsf{coMA})/1$. The proof is based on a similar lemma in a recent work [14] (which further builds on [30, 33]). We present a proof in Appendix A for completeness.

▶ **Lemma 10** (Implicit in [14]). *For all constants k, there is an integer c, and a language $L \in (MA \cap coMA)_{/1}$, such that for all sufficiently large $\tau \in \mathbb{N}$ and $n = 2^\tau$, either*

- *$SIZE(L_n) > n^k$, or*
- *$SIZE(L_m) > m^k$, for an $m \in (n^c, 2 \cdot n^c) \cap \mathbb{N}$.*

We also need the following two simple lemmas.

▶ **Lemma 11.** *NP is not in $SIZE[n^k]$ for all k iff $NP_{/n}$ is not in $SIZE[n^k]$ for all k.*

Proof. The $\Rightarrow$ direction is trivial. For the $\Leftarrow$ direction, suppose NP is in $\mathsf{SIZE}[n^k]$ for an integer k. Let $L \in \mathsf{NP}_{/n}$, and M and $\{\alpha_n\}_{n\in\mathbb{N}}$ be its corresponding nondeterministic Turing machine and advice sequence. Let $p(n)$ be a polynomial running time upper bound of M on inputs of length n.

Now, we define a language L' such that a pair $(x, \alpha) \in L'$ if and only if $|x| = |\alpha|$ and M accepts x with advice bits set to α in $p(|x|)$ steps. Clearly, $L' \in \mathsf{NP}$ from the definition, so it has an n^k-size circuit family. Fixing the advice bits to the actual α_n's in the circuit family, we have an $n^{O(k)}$-size circuit family for L as well. This completes the proof. ◀

▶ **Lemma 12** (Theorem 14 [18]). *Let k be an integer. If $\mathsf{NP} \subset \mathsf{P}_{/\mathsf{poly}}$ and all NP verifiers have n^k-size witnesses, then $\mathsf{NP} \subseteq \mathsf{MATIME}[n^{O(k)}] \subset \mathsf{SIZE}[n^{O(k)}]$.*

Proof. Assume all NP verifiers have n^k-size witnesses. By guessing circuits for the witnesses to PCP verifiers, it follows that $\mathsf{NP} \subseteq \mathsf{MATIME}[n^{O(k)}]$ [18]. Furthermore, we have $\mathsf{MATIME}[n^{O(k)}] \subset \mathsf{NTIME}[n^{O(k)}]_{/n^{O(k)}} \subset \mathsf{SIZE}[n^{O(k)}]$. The last step follows from the assumption that $\mathsf{NP} \subset \mathsf{P}_{/\mathsf{poly}}$ (and therefore $\mathsf{SAT} \in \mathsf{SIZE}[n^c]$ for a constant c). ◀

Now, we are ready to prove our equivalence theorem (restated below).

▶ **Reminder of Theorem 2.** *Assuming $\mathsf{NP} \subset \mathsf{P}_{/\mathsf{poly}}$, the following are equivalent:*

1. *NP is not in $\mathsf{SIZE}[n^k]$ for all k.*
2. *$\mathsf{AM}_{/1}$ is in c-r.o.-$\mathsf{NP}_{/n^\varepsilon}$ for all $\varepsilon > 0$ and integers c.*
3. *NP does not have n^k-size witnesses for all k.*[8]
4. *For all k and d, there is a poly-time nondeterministic PRG with n bits of advice against n^k-size circuits d-robustly often.*[9]
5. *NP is not in $\mathsf{AMTIME}(n^k)$ for all k.*
6. *$(\mathsf{NP} \cap \mathsf{coNP})_{/n^\varepsilon}$ is not in $\mathsf{SIZE}[n^k]$ for all k and all $\varepsilon > 0$.*
7. *$(\mathsf{AM} \cap \mathsf{coAM})_{/1}$ is in c-r.o.-$(\mathsf{NP} \cap \mathsf{coNP})_{/n^\varepsilon}$ for all $\varepsilon > 0$ and all integers c.*

Proof. We prove the following directions to show equivalence.

(2) ⇒ (1). Suppose (2) holds. For all k, let L and c be the $\mathsf{MA}_{/1}$ language and the corresponding constant c guaranteed by Lemma 10. By (2) and the fact that $\mathsf{MA}_{/1} \subseteq \mathsf{AM}_{/1}$, there is an $\mathsf{NP}_{/n}$ language L' such that for infinitely many n's, L' agrees with L on inputs with length in $[n, n^{2c}]$.

Let $\tau = \lceil \log(n) \rceil$. By the condition of Lemma 10, we know that for at least one $\ell \in [n, n^{2c}]$, we have $\mathsf{SIZE}(L'_\ell) \geq \ell^k$. Since there are infinitely many such n, we conclude that L' is not in $\mathsf{SIZE}[n^k]$. Since k can be an arbitrary integer, it further implies that $\mathsf{NP}_{/n}$ is not in $\mathsf{SIZE}[n^k]$ for all k, and hence also NP is not in $\mathsf{SIZE}[n^k]$ for all k by Lemma 11.

(1) ⇒ (3). We prove the contrapositive. Suppose NP has n^k-size witnesses for an integer k. Then, by Lemma 12, $\mathsf{NP} \subset \mathsf{SIZE}[n^{O(k)}]$.

(3) ⇒ (4). This more-or-less follows directly from standard hardness-to-pseudorandomness constructions [36]. More specifically, for all integers k and d and $\varepsilon > 0$, there is a language $L \in \mathsf{NP}$ without $n^{gkd/\varepsilon}$-size witnesses. Equivalently, there is a poly-time verifier V for L, such that there are infinitely many $x \in L$ such that for all y with $V(x, y) = 1$, it follows $CC(y) \geq |x|^{gkd/\varepsilon}$.

---

[8] See the statement of Theorem 2 in the introduction for the definition of n^k-size witnesses.
[9] See the Preliminaries for a full definition of nondeterministic PRG and d-robustly often.

For such an $x \in L$ with $|x| = m$, we can guess a y such that $V(x, y) = 1$ and apply Theorem 8 to construct a poly-time nondeterministic PRG with seed length $O(\log m)$, which works for input length $n \in [m^{1/\varepsilon}, m^{d/\varepsilon}]$ and against n^k-size circuits. Note that advice length is $|x| = m \leq n^\varepsilon$.

(4) ⇒ (2). First, under the assumption that $\mathsf{NP} \subset \mathsf{P}_{/\mathsf{poly}}$, we have the collapse $\mathsf{AM}_{/1} = \mathsf{MA}_{/1}$ [4]. So it suffices to show that $\mathsf{MA}_{/1} \subset$ c-r.o.-$\mathsf{NP}_{/n^\varepsilon}$ for all $\varepsilon > 0$ and integers d.

Let $L \in \mathsf{MA}_{/1}$. That is, for a constant k, there is an n^k-time algorithm $A(x, y, z, \alpha)$ with one bit of advice α_n, such that

- $x \in L \Rightarrow$ there is a y of $|x|^k$ length such that $\Pr_z[A(x, y, z, \alpha_n) = 1] \geq 2/3$.
- $x \notin L \Rightarrow$ for all y of $|x|^k$ length, $\Pr_z[A(x, y, z, \alpha_n) = 1] \leq 1/3$.

Fixing the x, y, α_n, we can construct a circuit $C_{x,y,\alpha_n}(z) := A(x, y, z, \alpha_n)$ of size n^{2k} in n^{2k} time.

Now, by (4), for all d, there is a poly-time NPRG G with seed length $O(\log n)$ and advice length n^ε such that there are infinitely many m's such that for all $n \in [m, m^d]$, G_n fools n^{2k}-size circuits.

Applying G_n to fool C_{x,y,α_n} directly, we have a language $L' \in \mathsf{NP}_{/n^\varepsilon}$ such that there are infinitely many m such that L' agrees with L on all input lengths in $[m, m^d]$. This completes the proof since d can be made arbitrarily large.

(5) ⇒ (3). We prove the contrapositive. Suppose NP has n^k-size witnesses for an integer k. By Lemma 12, it follows that $\mathsf{NP} \subseteq \mathsf{MATIME}[n^{O(k)}] \subseteq \mathsf{AMTIME}[n^{O(k)}]$.

(1) ⇒ (5). Again, we prove the contrapositive. We have $\mathsf{NP} \subseteq \mathsf{AMTIME}[n^{O(k)}] \subset \mathsf{NTIME}[n^{O(k)}]_{/n^{O(k)}} \subset \mathsf{SIZE}[n^{O(k)}]$. The last step follows from the assumption that $\mathsf{NP} \subseteq \mathsf{P}_{/\mathsf{poly}}$ (and therefore $\mathsf{SAT} \in \mathsf{SIZE}[n^c]$ for a constant c).

(6) ⇒ (1). $(\mathsf{NP} \cap \mathsf{coNP})_{/n^\varepsilon}$ is not in $\mathsf{SIZE}[n^k]$ for all k and $\varepsilon > 0$ implies $\mathsf{NP}_{/n}$ is not in $\mathsf{SIZE}[n^k]$ for all k, which in turn implies NP is not in $\mathsf{SIZE}[n^k]$ for all k by Lemma 11.

(4) ⇒ (7). This follows similarly as the direction from (4) to (2).

(7) ⇒ (6). This follows similarly as the direction from (2) to (1). Note that [4] also implies $(\mathsf{MA} \cap \mathsf{coMA})_{/1} = (\mathsf{AM} \cap \mathsf{coAM})_{/1}$ under the assumption $\mathsf{NP} \subset \mathsf{P}_{/\mathsf{poly}}$. ◀

5 NP Circuit Lower Bounds Imply Better Karp-Lipton Collapses

Now we show a corollary of Theorem 2 that NP circuit lower bounds imply better Karp-Lipton collapses.

▶ **Reminder of Theorem 4.** *Let $\mathcal{C} \in \{\oplus P, PSPACE, PP, EXP\}$. Suppose $NP \not\subset SIZE[n^k]$ for all k. Then for all $\varepsilon > 0$, $(\mathcal{C} \subset P_{/\mathsf{poly}} \implies \mathcal{C} \subset i.o.\text{-}NP_{/n^\varepsilon})$. In particular, polynomial-size circuits for any $\mathcal{C}$-complete language can be constructed in NP on infinitely many input lengths with n^ε advice.*

Proof of Theorem 4. We first prove it for $\oplus\mathsf{P}$. Suppose for all k, $\mathsf{NP} \not\subset \mathsf{SIZE}[n^k]$ and $\oplus\mathsf{P} \subset \mathsf{P}_{/\mathsf{poly}}$.

First, note that $\mathsf{BPP}^{\oplus\mathsf{P}} \subset \mathsf{P}_{/\mathsf{poly}}$, implying $\mathsf{PH} \subset \mathsf{P}_{/\mathsf{poly}}$ by Toda's theorem [34]. Therefore, by Theorem 2 together with our assumption, we have $\mathsf{MA} \subset$ c-r.o.-$\mathsf{NP}_{/n^\varepsilon}$ for all $\varepsilon > 0$ and integers c. In particular, $\mathsf{MA} \subset$ i.o.-$\mathsf{NP}_{/n^\varepsilon}$ for all $\varepsilon > 0$. Now it suffices to show that $\oplus\mathsf{P} \subset \mathsf{P}_{/\mathsf{poly}} \implies \oplus\mathsf{P} \subseteq \mathsf{MA}$.

Let Π be the random self-reducible and downward self-reducible $\oplus\mathsf{P}$-complete language in [23]. By our assumption that $\oplus\mathsf{P} \subset \mathsf{P}_{/\mathsf{poly}}$, Π has a poly-size circuit family.

Then we can guess-and-verify these circuits in MA. We first existentially guess a circuit C_k for Π on every input length $k = 1, \ldots, n$. C_1 can be verified in constant time, and each successive circuit can be verified via random downward self-reducibility: given a circuit of length m that computes Π_m exactly, a circuit of length $m+1$ can be checked on random inputs to verify (with high probability) its consistency with Π_{m+1} (which is computable using the downward self-reducibility and the circuit for Π_m). Then we can apply the random self-reducibility to construct an exact circuit for Π_{m+1} from C_{m+1} with high probability, as we already know C_{m+1} approximates Π_{m+1} very well. Therefore, with high probability, we can guess-and-verify a circuit for Π_n via a poly-time MA computation. This puts $\oplus\mathsf{P} \subseteq \mathsf{MA}$. Combining that with $\mathsf{MA} \subset \text{i.o.-}\mathsf{NP}_{/n^\varepsilon}$ for all $\varepsilon > 0$, we can conclude that $\oplus\mathsf{P} \subset \text{i.o.-}\mathsf{NP}_{/n^\varepsilon}$ for all $\varepsilon > 0$.

To construct a circuit for Π_n in i.o.-$\mathsf{NP}_{/n^\varepsilon}$, note that by Theorem 6, for all k, we have an i.o.-NPRG fooling n^k-size circuits. We can pick k to be a sufficiently large integer, and use the i.o.-NPRG to derandomize the above process. This turns out to be more subtle than one might expect.

Construction of poly-size circuits of Π_n in i.o.-$\mathsf{NP}_{/n^\varepsilon}$. Let d be a sufficiently large constant. Since we only aim for an i.o.-construction, we can assume that our i.o.-NPRG works for the parameter n, and fools all n^d-size circuits. Also, suppose we have $\mathsf{SIZE}(\Pi_n) \leq n^c$ for all n and a constant c.

We say a circuit C γ-approximates a function f, if $C(x) = f(x)$ for at least a γ fraction of the inputs.

Again, suppose we already constructed the circuits $C_1, C_2, \ldots, C_k$ for $\Pi_1, \Pi_2, \ldots, \Pi_k$. This time we cannot guarantee C_i exactly computes Π_i. Instead, we relax the condition a bit and ensure that C_i $(1 - 4/n)$-approximates Π_i for all $i \in [k]$. Clearly, we can check $C_1 \equiv \Pi_1$ directly so this can be satisfied when $k = 1$.

We now show how to construct an approximate circuit for Π_{k+1}. First, using the random self-reducibility of Π and the circuit C_k approximating Π_k, there is an oracle circuit E of size $\mathrm{poly}(n)$, which takes two inputs x with $|x| = k$ and r with $|r| = \mathrm{poly}(n)$, such that for all x,

$$\Pr_r \left[E^{C_k}(x, r) = \Pi_k(x)\right] \geq 1 - 1/2^n.$$

Also, by the downward self-reducibility of Π, there is an oracle machine D of $\mathrm{poly}(k)$ size, such that $D^{\Pi_k}(z) = \Pi_{k+1}(z)$ for all z.

Now, consider the following circuit $G(x, r)$ for computing Π_{k+1}: the circuit simulates D^{Π_k}, while answering all queries w to Π_k using $E^{C_k}(w, r)$. For each input $x \in \{0,1\}^{k+1}$, let $w_1, w_2, \ldots, w_{\mathrm{poly}(n)}$ be all queries to Π_k made by running D on the input x assuming all answers are correct, we can see that if $E^{C_k}(w_j, r) = \Pi_k(w_j)$ for all these w_j's, then $G(x, r) = \Pi_{k+1}(x)$. Therefore, we have

$$\Pr_r[G(x, r) = \Pi_{k+1}(x)] \geq 1 - \mathrm{poly}(n)/2^n,$$

for all $x \in \{0,1\}^{k+1}$.

Now, we guess a circuit C_{k+1} of size $(k+1)^c$ which is supposed to compute Π_{k+1}. By an enumeration of all possible seeds to our NPRG, we can estimate the probability

$$p_{\mathsf{good}} := \Pr_{x \in \{0,1\}^{k+1}} \Pr_r[G(x, r) = C_{k+1}(x)].$$

within $1/n$ in $\mathrm{poly}(n)$ time, as the expression $[G(x, r) = C_{k+1}(x)]$ has a $\mathrm{poly}(n)$ size circuit with inputs being x and r. Let our estimation be p_{est}. We have $|p_{\mathsf{good}} - p_{\mathsf{est}}| \leq 1/n$.

Putting the above together, we have

$$\left| \Pr_{x \in \{0,1\}^{k+1}} [\Pi_{k+1}(x) = C_{k+1}(x)] - p_{\mathsf{good}} \right| \leq \mathrm{poly}(n)/2^n.$$

We reject immediately if our estimation $p_{\mathsf{est}} < 1 - 2/n$ (note that if C_{k+1} is the correct circuit, p_{good} would be larger than $1 - \mathrm{poly}(n)/2^n > 1 - 1/n$, and therefore $p_{\mathsf{est}} > 1 - 2/n$). So after that, we can safely assume that C_{k+1} $(1 - 4/n)$-approximates Π_{k+1}.

Therefore, at the end we have an n^c-size circuit C_n which $(1-4/n)$-approximates Π_n, and we try to recover an exact circuit for Π_n from C_n by exploiting the random self-reducibility of Π_n again. Note that there is an oracle circuit $E(x, r)$, which takes two inputs x with $|x| = n$ and r with $|r| = \mathrm{poly}(n)$ such that for all x,

$$\Pr_r[E^{C_n}(x, r) = \Pi_n(x)] \geq 2/3.$$

Now, we generate $\ell = n^{O(1)}$ strings $r_1, r_2, \ldots, r_\ell$ by enumerating all seeds to the NPRG. We construct our final circuit C to be the majority of $E^{C_n}(x, r_j)$ for all $j \in [\ell]$. It is not hard to see that C computes Π_n exactly, as our inputs $\{r_j\}_{j \in [\ell]}$ fool the expression $\left[E^{C_n}(x, r) = \Pi_n(x)\right]$ for all $x \in \{0,1\}^n$.

For the case of PP and PSPACE, one can implement the above procedure in the same way, using the corresponding random self-reducible and downward self-reducible PP-complete and PSPACE-complete languages (Permanent and the PSPACE-complete language in [35]).

For the case of EXP, note that $\mathsf{EXP} \subset \mathsf{P}_{/\mathsf{poly}} \implies \mathsf{EXP} = \mathsf{PSPACE}$, so we can proceed the same way as for PSPACE (since EXP = PSPACE, PSPACE-complete languages are also EXP-complete). ◀

6 Consequence of Weak Circuit Lower Bounds for Sparse Languages in NP

Now, we are ready to prove the consequences of weak circuit lower bounds for sparse NP languages. We first need the following lemma.

▶ **Lemma 13** (Hardness Verification from Circuit Lower Bounds for Sparse NTIME$[T(n)]$ Languages). *Let $S_{ckt}(n), S_{sparse}(n), T(n) : \mathbb{N} \to \mathbb{N}$ be time constructible functions. Suppose there is an $S_{sparse}(n)$-sparse language $L \in$ NTIME$[T(n)]$ without $(n \cdot S_{ckt}(n))$-size circuits. Then there is a procedure V such that:*

- *V takes a string z of length $n \cdot S_{sparse}(n)$ as input and an integer $\ell \leq S_{sparse}(n)$ as advice.*
- *V runs in $O(S_{sparse}(n) \cdot T(n))$ nondeterministic time.*
- *For infinitely many n, there is an integer $\ell_n \leq S_{sparse}(n)$ such that $V(z, \ell_n)$ accepts exactly one string z, and z has circuit complexity $\Omega(S_{ckt}(n)/\log S_{sparse}(n))$.*

Proof. Let L be the NTIME$[T(n)]$ language in the assumption. Let $N = n \cdot S_{\mathsf{sparse}}(n)$. We define a string $\mathsf{List}_{L_n} \in \{0,1\}^N$ as the concatenation of all $x \in L_n$ in lexicographical order, together with additional zeros at the end to make the string have length exactly N.

Now define a function f_n on $m = \log\lceil N + 1 \rceil$ bits, with truth-table $\mathsf{List}_{L_n} 10^{2^m - N}$.

We claim that $\mathsf{SIZE}(L_n) \leq O(\mathsf{SIZE}(f_n) \cdot n \cdot \log(S_{\mathsf{sparse}}(n)))$. To determine whether $x \in L_n$, it would suffice to perform a binary search on the list List_{L_n}. We construct a circuit for L_n which performs binary search using f_n. First, we hard-wire the length of the list $\ell := |L_n| \leq S_{\mathsf{sparse}}(n)$ into our circuit for L_n so that the binary search can begin with the correct range. A binary search on $\mathsf{List}(L_n)$ takes $O(\log S_{\mathsf{sparse}}(n))$ comparisons, and each

comparison requires $O(n)$ calls to f_n (to print the appropriate string). It is easy to see that the circuit size required for the binary search is dominated by the total cost of the comparisons; this proves the claim.

From the assumption, we know that for infinitely many n, L_n has no circuit of size $n \cdot S_{\mathsf{ckt}}(n)$. By our upper bound on the circuit size of L_n, it follows that on the same set of n, the function f_n has circuit complexity at least $\Omega(S_{\mathsf{ckt}}(n)/\log S_{\mathsf{sparse}}(n))$.

Now, we construct an algorithm V that only accepts the string $f_n = \mathsf{List}_{L_n} 10^{2^m - N}$. We first need the integer $\ell = |L_n|$ as the advice. Given a string Y of length N, we check that Y contains exactly ℓ distinct inputs in $\{0,1\}^n$ in lexicographical order with the correct format, and we guess an $O(T(n))$-length witness for each input to verify they are indeed all in L. It is easy to see that V runs in $O(S_{\mathsf{sparse}}(n) \cdot T(n))$ nondeterministic time, which completes the proof. ◀

▶ Remark 14. Note that the advice integer ℓ can be calculated directly with an NP oracle by doing a binary search for ℓ, which takes $O(\log S_{\mathsf{sparse}}(n))$ NP-oracle calls. That is, one can also use a $\mathsf{P}^{\mathsf{NP}[O(\log S_{\mathsf{sparse}}(n))]}$ verifier without advice bits in the statement of Lemma 13.

▶ Remark 15. As mentioned in the introduction, the above proof can be seen as a type of *hardness condensation* for all sparse $\mathsf{NTIME}[T(n)]$ languages. The goal of hardness condensation [8, 22] is that, given a hard function f on n input bits with complexity S, we want to construct a function $\widetilde{f}$ on $\ell \ll n$ input bits that still has complexity roughly S. The above proof shows any hard sparse language in $\mathsf{NTIME}[T(n)]$ can be "condensed" into a function representing its sorted yes-instances.

Combing Lemma 13 with Theorem 8, we obtain a construction of an i.o.-NPRG.

▶ **Corollary 16** (NPRG from lower bounds against sparse $\mathsf{NTIME}[T(n)]$ languages)**.** *Under the circuit lower bound assumption of Lemma 13, there is an i.o.-NPRG G with the properties:*

- *G has $O(\log S_{\mathsf{sparse}}(n) + \log n)$ seed length.*
- *G takes $O(\log S_{\mathsf{sparse}}(n))$ bits of advice.*
- *G runs in $S_{\mathsf{sparse}}(n) \cdot T(n) + \mathrm{poly}(n \cdot S_{\mathsf{sparse}}(n))$ time.*
- *G fools circuits of size at most $(S_{\mathsf{ckt}}(n)/\log S_{\mathsf{sparse}}(n))^{\Omega(1)}$.*

Now we are ready to prove Theorem 6.

▶ **Reminder of Theorem 6.** *Suppose there is an $\varepsilon > 0$, a $c \geq 1$, and an n^c-sparse $L \in \mathsf{NP}$ without $n^{1+\varepsilon}$-size circuits. Then $\mathsf{MA} \subset$ i.o.-$\mathsf{NP}_{/O(\log n)}$, $\mathsf{MA} \subseteq$ i.o.-$\mathsf{P}^{\mathsf{NP}[O(\log n)]}$, and $\mathsf{NE} \not\subset \mathsf{SIZE}[2^{\delta \cdot n}]$ for some $\delta > 0$ (which implies $\mathsf{NP} \not\subset \mathsf{SIZE}[n^k]$ for all k).*

Proof. First, by Corollary 16 and setting $S_{\mathsf{ckt}}(n) = n^\varepsilon$, $S_{\mathsf{sparse}}(n) = n^c$ and $T(n) = \mathrm{poly}(n)$, there is an i.o.-NPRG with seed length $O(\log n)$ which takes $O(\log n)$ bits of advice, runs in $\mathrm{poly}(n)$ time, and fools circuits of size $n^{\Omega(\varepsilon)} = n^{\Omega(1)}$. Note that we can simply scale it up to make it fool circuits of size n^k for any k, with only a constant factor blowup on seed length and advice bits and a polynomial blowup on the running time.

Applying the i.o.-NPRG to arbitrary Merlin-Arthur computations, we conclude $\mathsf{MA} \subset$ i.o.-$\mathsf{NP}_{/O(\log n)}$. Similarly, $\mathsf{MA} \subseteq$ i.o.-$\mathsf{P}^{\mathsf{NP}[O(\log n)]}$ follows from Remark 14.

Now we show $\mathsf{NE} \not\subset \mathsf{SIZE}[2^{\delta \cdot n}]$ for some $\delta > 0$. By Lemma 13, there is a nondeterministic algorithm running in $\mathrm{poly}(n)$ time, given $\alpha_n = c \log n$ bits of advice, guess and verify a string of length n^{c+1} which has circuit complexity at least $n^{\varepsilon/2}$, for infinitely many n. We say these infinitely many n are *good* n.

Next, we define the following language $L \in \mathsf{NE}$: Given an input of length m. It treats the first $\ell = m/4c$ bits a binary encoded integer $n \leq 2^\ell$. Then it treats the next $c \log n$ input bits a as the advice, and tries to guess-and-verify a string z which passes the verification procedure in Lemma 13 with advice a and parameter n, and then it treats the next $(c+1) \cdot \log n$ input bits as an integer $i \in [n^{c+1}]$, and accepts if and only $z_i = 1$.

First, it is easy to verify $L \in \mathsf{NE}$, as the algorithm runs in $\mathrm{poly}(n) = 2^{O(\ell)} = 2^{O(m)}$ nondeterministic time. For the circuit complexity of L, we know that for the good n, on inputs of length of $m = 4 \cdot c \cdot \lceil \log n \rceil$, if we fix the first $m/4c$ bits to represent the integer n, and next $c \log n$ bit to the actual advice α_n, L would compute the hard string of length n^{c+1} on the next $(c+1) \cdot \log n$ bits. Therefore, $\mathsf{SIZE}(L_m) \geq n^\varepsilon \geq 2^{\Omega(m)}$ for infinitely many m's, which completes the proof. ◀

Finally, we prove Theorem 7.

▶ **Reminder of Theorem 7.** *NEXP $\not\subset$ P$_{/\mathsf{poly}}$ if and only if there is an $\varepsilon > 0$ such that for all sufficiently small $\beta > 0$, there is a 2^{n^β}-sparse language $L \in$ NTIME$[2^{n^\beta}]$ without $n^{1+\varepsilon}$-size circuits.*

Proof. ($\Rightarrow$) This direction is easy to prove using standard methods. Suppose $\mathsf{NEXP} \not\subset \mathsf{P}_{/\mathsf{poly}}$; this also implies $\mathsf{NE} \not\subset \mathsf{P}_{/\mathsf{poly}}$. Therefore, there is a language $L \in \mathsf{NTIME}[2^n]$ that does not have $n^{2/\beta}$-size circuits. Define a padded language $L' = \{x10^{|x|^{1/\beta}-1} | x \in L\}$. It is easy to see that $L' \in \mathsf{NTIME}[2^{m^\beta}]$, by running the NE algorithm for L on its first $n = O(m^\beta)$ input bits. From the circuit lower bound on L, it follows that L' does not have $n^{2/\beta} = m^2$-size circuits.

($\Leftarrow$) First, by Impagliazzo-Kabanets-Wigderson [24], if for every ε and integer k, there is an i.o.-NPRG with seed length n^ε, n^ε advice bits, and 2^{n^ε} running time that fools n^k-size circuits, then $\mathsf{NEXP} \not\subset \mathsf{P}_{/\mathsf{poly}}$.

Setting $S_{\mathsf{ckt}}(n) = n^\varepsilon$, $S_{\mathsf{sparse}}(n) = 2^{n^\beta}$ and $T(n) = 2^{n^\beta}$ in Corollary 16, there is an i.o.-NPRG with seed length $O(n^\beta)$, takes $O(n^\beta)$ bits of advice, and runs in $2^{O(n^\beta)}$ time that fools circuits of size $n^{\Omega(\varepsilon/\beta)} = n^{\varepsilon'}$ for $\varepsilon' > 0$. By setting $m = n^{\varepsilon'/k}$, we obtain an i.o.-NPRG with seed/advice length $O(m^{\beta \cdot k/\varepsilon'})$ and running time $2^{O(m^{\beta \cdot k/\varepsilon'})}$, which fools circuits of size m^k. Therefore, by [24], it follows that $\mathsf{NEXP} \not\subset \mathsf{P}_{/\mathsf{poly}}$. ◀

7 Open Problems

We conclude with three interesting open questions stemming from this work.

1. *Are fixed-polynomial circuit lower bounds for NP equivalent to a Karp-Lipton collapse of PH to NP?*
   Formally, is $\mathsf{NP} \not\subset \mathsf{SIZE}[n^k]$ for all k equivalent to $(\mathsf{NP} \subset \mathsf{P}_{/\mathsf{poly}} \implies \mathsf{PH} \subset \text{i.o.-}\mathsf{NP}_{/n})$? Recall we showed that similar Karp-Lipton-style collapses do occur, assuming NP circuit lower bounds (e.g., $(\mathsf{PSPACE} \subset \mathsf{P}_{/\mathsf{poly}} \implies \mathsf{PSPACE} \subset \text{i.o.-}\mathsf{NP}_{/n})$), and we showed that $\mathsf{NP} \not\subset \mathsf{SIZE}[n^k]$ implies a type of collapse of AM into NP.
2. It is also a prominent open problem to prove that $\mathsf{ZPP}^{\mathsf{NP}}_{tt} \not\subset \mathsf{SIZE}[n^k]$ for some constant k [16] (that is, prove lower bounds for ZPP with nonadaptive queries to an NP oracle). *Is this lower bound equivalent to a Karp-Lipton collapse of PH?*
   The difficulty is that, assuming $\mathsf{ZPP}^{\mathsf{NP}}_{tt} \not\subset \mathsf{SIZE}[n^k]$, it appears that we may obtain a good simulation of $\mathsf{BPP}^{\mathsf{NP}}_{tt}$, but we presently have no Karp-Lipton Theorem collapsing PH to $\mathsf{BPP}^{\mathsf{NP}}_{tt}$ (indeed, lower bounds for this class are also open). Furthermore, [16] observe that $\mathsf{NP} \subset \mathsf{P}/\mathrm{poly}$ does imply the (small) collapse $\mathsf{BPP}^{\mathsf{NP}}_{tt} = \mathsf{ZPP}^{\mathsf{NP}}_{tt}$; it is unclear how a circuit lower bound against $\mathsf{ZPP}^{\mathsf{NP}}_{tt}$ would aid a further collapse.

3. In light of our Theorem 7, *is it possible to show interesting hardness magnification results for non-sparse versions of* MCSP *(say,* MCSP$[2^m/m^2]$*)?*
   Currently, we only know hardness magnification results when the circuit size parameter is $2^{o(m)}$ [32, 31, 29].

References

1 Scott Aaronson. Oracles Are Subtle But Not Malicious. In *21st Annual IEEE Conference on Computational Complexity (CCC 2006), 16-20 July 2006, Prague, Czech Republic*, pages 340–354, 2006. doi:10.1109/CCC.2006.32.

2 Eric Allender and Michal Koucký. Amplifying lower bounds by means of self-reducibility. *J. ACM*, 57(3):14:1–14:36, 2010. doi:10.1145/1706591.1706594.

3 Sanjeev Arora and Boaz Barak. *Computational Complexity - A Modern Approach*. Cambridge University Press, 2009. URL: http://www.cambridge.org/catalogue/catalogue.asp?isbn=9780521424264.

4 Vikraman Arvind, Johannes Köbler, Uwe Schöning, and Rainer Schuler. If NP has Polynomial-Size Circuits, then MA=AM. *Theor. Comput. Sci.*, 137(2):279–282, 1995. doi:10.1016/0304-3975(95)91133-B.

5 László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. Checking Computations in Polylogarithmic Time. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing, May 5-8, 1991, New Orleans, Louisiana, USA*, pages 21–31, 1991. doi:10.1145/103418.103428.

6 Nader H. Bshouty, Richard Cleve, Ricard Gavaldà, Sampath Kannan, and Christino Tamon. Oracles and Queries That Are Sufficient for Exact Learning. *J. Comput. Syst. Sci.*, 52(3):421–433, 1996. doi:10.1006/jcss.1996.0032.

7 Harry Buhrman and Steven Homer. Superpolynomial Circuits, Almost Sparse Oracles and the Exponential Hierarchy. In *Foundations of Software Technology and Theoretical Computer Science, 12th Conference, New Delhi, India, December 18-20, 1992, Proceedings*, pages 116–127, 1992. doi:10.1007/3-540-56287-7_99.

8 Joshua Buresh-Oppenheim and Rahul Santhanam. Making Hard Problems Harder. In *21st Annual IEEE Conference on Computational Complexity (CCC 2006), 16-20 July 2006, Prague, Czech Republic*, pages 73–87, 2006. doi:10.1109/CCC.2006.26.

9 Jin-yi Cai. $\mathsf{S}_2^{\mathsf{P}}$ is subset of $\mathsf{ZPP}^{\mathsf{NP}}$. *J. Comput. Syst. Sci.*, 73(1):25–35, 2007. doi:10.1016/j.jcss.2003.07.015.

10 Jin-yi Cai, Venkatesan T. Chakaravarthy, Lane A. Hemaspaandra, and Mitsunori Ogihara. Competing provers yield improved Karp-Lipton collapse results. *Inf. Comput.*, 198(1):1–23, 2005. doi:10.1016/j.ic.2005.01.002.

11 Jin-yi Cai and Osamu Watanabe. On Proving Circuit Lower Bounds against the Polynomial-Time Hierarchy. *SIAM J. Comput.*, 33(4):984–1009, 2004. doi:10.1137/S0097539703422716.

12 Venkatesan T Chakaravarthy and Sambuddha Roy. Oblivious symmetric alternation. In *Annual Symposium on Theoretical Aspects of Computer Science*, pages 230–241. Springer, 2006.

13 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.

14 Lijie Chen. Non-deterministic Quasi-Polynomial Time is Average-case Hard for ACC Circuits. *Electronic Colloquium on Computational Complexity (ECCC)*, 26:31, 2019. URL: https://eccc.weizmann.ac.il/report/2019/031.

15 Lijie Chen and Roei Tell. Bootstrapping results for threshold circuits "just beyond" known lower bounds. *Electronic Colloquium on Computational Complexity (ECCC)*, 25:199, 2018. URL: https://eccc.weizmann.ac.il/report/2018/199.

16 Peter Dixon, Aduri Pavan, and N. V. Vinodchandran. On Pseudodeterministic Approximation Algorithms. In *43rd International Symposium on Mathematical Foundations of Computer Science, MFCS 2018, August 27-31, 2018, Liverpool, UK*, pages 61:1–61:11, 2018. doi:10.4230/LIPIcs.MFCS.2018.61.

17 Lance Fortnow and Rahul Santhanam. Robust simulations and significant separations. *Inf. Comput.*, 256:149–159, 2017. doi:10.1016/j.ic.2017.07.002.

18 Lance Fortnow, Rahul Santhanam, and Ryan Williams. Fixed-Polynomial Size Circuit Bounds. In *Proceedings of the 24th Annual IEEE Conference on Computational Complexity, CCC 2009, Paris, France, 15-18 July 2009*, pages 19–26, 2009. doi:10.1109/CCC.2009.21.

19 Oded Goldreich. *Computational complexity - a conceptual perspective*. Cambridge University Press, 2008.

20 Juris Hartmanis, Neil Immerman, and Vivian Sewelson. Sparse Sets in NP-P: EXPTIME versus NEXPTIME. *Information and Control*, 65(2/3):158–181, 1985. doi:10.1016/S0019-9958(85)80004-8.

21 Russell Impagliazzo, 2018. Personal Communication.

22 Russell Impagliazzo, Ragesh Jaiswal, Valentine Kabanets, and Avi Wigderson. Uniform Direct Product Theorems: Simplified, Optimized, and Derandomized. *SIAM J. Comput.*, 39(4):1637–1665, 2010. doi:10.1137/080734030.

23 Russell Impagliazzo, Valentine Kabanets, and Ilya Volkovich. The Power of Natural Properties as Oracles. In *33rd Computational Complexity Conference, CCC 2018, June 22-24, 2018, San Diego, CA, USA*, pages 7:1–7:20, 2018. doi:10.4230/LIPIcs.CCC.2018.7.

24 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.

25 Ravi Kannan. Circuit-Size Lower Bounds and Non-Reducibility to Sparse Sets. *Information and Control*, 55(1-3):40–56, 1982. doi:10.1016/S0019-9958(82)90382-5.

26 Richard Karp and Richard Lipton. Turing Machines That Take Advice. *L'Enseignement Mathématique*, 28(2):191–209, 1982.

27 Johannes Köbler and Osamu Watanabe. New Collapse Consequences of NP Having Small Circuits. *SIAM J. Comput.*, 28(1):311–324, 1998. doi:10.1137/S0097539795296206.

28 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.

29 Dylan McKay, Cody Murray, and Ryan Williams. Weak Lower Bounds on Resource-Bounded Compression Imply Strong Separations of Complexity Classes, 2019. To appear in STOC 2019.

30 Cody Murray and R. 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, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 890–901, 2018. doi:10.1145/3188745.3188910.

31 Igor Carboni Oliveira, Ján Pich, and Rahul Santhanam. Hardness magnification near state-of-the-art lower bounds, 2019. To appear in CCC 2019.

32 Igor Carboni Oliveira and Rahul Santhanam. Hardness Magnification for Natural Problems. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 65–76, 2018. doi:10.1109/FOCS.2018.00016.

33 Rahul Santhanam. Circuit Lower Bounds for Merlin–Arthur Classes. *SIAM J. Comput.*, 39(3):1038–1061, 2009. doi:10.1137/070702680.

34 Seinosuke Toda. PP is as hard as the polynomial-time hierarchy. *SIAM J. Comput.*, 20(5):865–877, 1991. doi:10.1137/0220053.

35 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.

36 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.

37 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.

38 Chee-Keng Yap. Some Consequences of Non-Uniform Conditions on Uniform Classes. *Theor. Comput. Sci.*, 26:287–300, 1983. doi:10.1016/0304-3975(83)90020-8.

A Almost Almost-everywhere $(\mathsf{MA} \cap \mathsf{coMA})_{/1}$ Circuit Lower Bounds

Here we provide a proof for Lemma 10 for completeness. The proof is based on a similar lemma from [14].

A.1 Preliminaries

A crucial ingredient of the proof is a PSPACE-complete language [35] satisfying strong reducibility properties, which is also used in the fixed-polynomial lower bounds for $\mathsf{MA}_{/1}$ and promiseMA [33], and the recent new witness lemmas for NQP and NP [30].

We first define these reducibility properties.

▶ **Definition 17.** *Let $L : \{0,1\}^* \to \{0,1\}$ be a language, we define the following properties:*

- *L is* downward self-reducible *if there is a constant c such that for all sufficiently large n, there is an n^c size uniform oracle circuit A such that for all $x \in \{0,1\}^n$, $A^{L_{n-1}}(x) = L_n(x)$.*
- *L is* paddable*, if there is a polynomial time computable projection Pad (that is, each output bit is either a constant or only depends on 1 input bit), such that for all integers $1 \leq n < m$ and $x \in \{0,1\}^n$, we have $x \in L$ if and only if $\mathsf{Pad}(x, 1^m) \in L$, where $\mathsf{Pad}(x, 1^m)$ always has length m.*
- *L is* same-length checkable *if there is a probabilistic polynomial-time oracle Turing machine M with output in $\{0,1,?\}$, such that, for any input x,*
  - *M asks its oracle queries only of length $|x|$.*
  - *If M is given L as an oracle, then M outputs $L(x)$ with probability 1.*
  - *M outputs $1 - L(x)$ with probability at most $1/3$ no matter which oracle is given to it.*

  We call M an instance checker *for L.*

▶ Remark 18. Note that the paddable property implies that $\mathsf{SIZE}(L_n)$ is non-decreasing.

The following PSPACE-complete language is given by [33] (modifying a construction of Trevisan and Vadhan [35]).

▶ **Theorem 19** ([33, 35])**.** *There is a PSPACE-complete language L^{PSPACE} which is paddable, downward self-reducible, and same-length checkable.*

We also need the following folklore theorem which is proved by a direct diagonalization against all small circuits.

▶ **Theorem 20.** *Let $n \leq s(n) \leq 2^{o(n)}$ be space-constructible. There is a universal constant c and a language $L \in \mathsf{SPACE}[s(n)^c]$ that $\mathsf{SIZE}(L_n) > s(n)$ for all sufficiently large n.*

A.2 Definitions

We need the following convenient definition of an $\mathsf{MA} \cap \mathsf{coMA}$ algorithm, which simplifies the presentation.

▶ **Definition 21.** *A language L is in $\mathsf{MA} \cap \mathsf{coMA}$, if there is a deterministic algorithm $A(x, y, z)$ (which is called the predicate) such that:*

- *A takes three inputs x, y, z such that $|x| = n$, $|y| = |z| = \mathrm{poly}(n)$ (y is the witness while z is the collection of random bits), runs in $O(T(n))$ time, and outputs an element from $\{0,1,?\}$.*

- *(Completeness) There exists a y such that*

$$\Pr_z[A(x,y,z) = L(x)] \geq 2/3.$$

- *(Soundness) For all y,*

$$\Pr_z[A(x,y,z) = 1 - L(x)] \leq 1/3.$$

▶ Remark 22. $(\mathsf{MA} \cap \mathsf{coMA})$ languages with advice are defined similarly, with A being an algorithm with the corresponding advice.

Note that by above definition, the semantic of $(\mathsf{MA} \cap \mathsf{coMA})_{/1}$ is different from $\mathsf{MA}_{/1} \cap \mathsf{coMA}_{/1}$. A language in $(\mathsf{MA} \cap \mathsf{coMA})_{/1}$ has both an $\mathsf{MA}_{/1}$ algorithm and a $\mathsf{coMA}_{/1}$ algorithm, and *their advice bits are the same*. While a language in $\mathsf{MA}_{/1} \cap \mathsf{coMA}_{/1}$ can have an $\mathsf{MA}_{/1}$ algorithm and a $\mathsf{coMA}_{/1}$ algorithm with different advice sequences.

A.3 Proof for Lemma 10

Now we are ready to prove Lemma 10 (restated below).

▶ **Reminder of Lemma 10.** *For all constants k, there is an integer c, and a language $L \in (\mathsf{MA} \cap \mathsf{coMA})_{/1}$, such that for all sufficiently large $\tau \in \mathbb{N}$ and $n = 2^\tau$, either*

- $\mathsf{SIZE}(L_n) > n^k$, *or*
- $\mathsf{SIZE}(L_m) > m^k$, *for an $m \in (n^c, 2 \cdot n^c) \cap \mathbb{N}$.*

Proof. Let L^{PSPACE} be the language specified by Theorem 19. By Theorem 20, there is an integer c_1 and a language L^{diag} in $\mathsf{SPACE}(n^{c_1})$, such that $\mathsf{SIZE}(L_n^{\mathsf{diag}}) \geq n^k$ for all sufficiently large n. By the fact that L^{PSPACE} is PSPACE-complete, there is a constant c_2 such that L_n^{diag} can be reduced to L^{PSPACE} on input length n^{c_2} in n^{c_2} time. We set $c = c_2$.

The Algorithm. Let $\tau \in \mathbb{N}$ be sufficiently large. We also let b to be a constant to be specified later. Given an input x of length $n = 2^\tau$ and let $m = n^c$, we first provide an informal description of the $(\mathsf{MA} \cap \mathsf{coMA})_{/1}$ algorithm which computes the language L. There are two cases:

1. When $\mathsf{SIZE}(L_m^{\mathsf{PSPACE}}) \leq n^b$. That is, when L_m^{PSPACE} is *easy*. In this case, on inputs of length n, we guess-and-verify a circuit for L_m^{PSPACE} of size n^b and use that to compute L_n^{diag}.
2. Otherwise, we know L_m^{PSPACE} is *hard*. Let ℓ be the largest integer such that $\mathsf{SIZE}(L_\ell^{\mathsf{PSPACE}}) \leq n^b$. On inputs of length $m_1 = m + \ell$, we guess-and-verify a circuit for L_ℓ^{PSPACE} and compute it (that is, compute L_ℓ^{PSPACE} on the first ℓ input bits while ignoring the rest).

Intuitively, the above algorithm computes a hard function because either it computes the hard language L_n^{diag} on inputs of length n, or it computes the hard language L_ℓ^{PSPACE} on inputs of length m_1. A formal description of the algorithm is given in Algorithm 1, while an algorithm for setting the advice sequence is given in Algorithm 2. It is not hard to see that a y_n can only be set once in Algorithm 2.

Algorithm 1: The $\mathsf{MA} \cap \mathsf{coMA}$ algorithm.

Given an input x with input length $n = |x|$;
Given an advice bit $y = y_n \in \{0, 1\}$;
Let $m = n^c$;
Let $n_0 = n_0(n)$ be the largest integer such that $n_0^c \leq n$;
Let $m_0 = n_0^c$;
Let $\ell = n - m_0$;
if $y = 0$ **then**
    Output 0 and terminate
if *n is a power of* 2 **then**
    (We are in the case that $\mathsf{SIZE}(L_m^{\mathsf{PSPACE}}) \leq n^b$.);
    Compute z in n^c time such that $L_n^{\mathsf{diag}}(x) = L_m^{\mathsf{PSPACE}}(z)$;
    Guess a circuit C of size at most n^b;
    Let M be the instance checker for L_m^{PSPACE};
    Flip an appropriate number of random coins, let them be r;
    Output $M^C(z, r)$;
else
    (We are in the case that $\mathsf{SIZE}(L_{m_0}^{\mathsf{PSPACE}}) > n_0^b$ and ℓ is the largest integer such that $\mathsf{SIZE}(L_\ell^{\mathsf{PSPACE}}) \leq n_0^b$.);
    Let z be the first ℓ bits of x;
    Guess a circuit C of size at most n_0^b;
    Let M be the instance checker for L_ℓ^{PSPACE};
    Flip an appropriate number of random coins, let them be r;
    Output $M^C(z, r)$;

Algorithm 2: The algorithm for setting advice bits.

All y_n's are set to 0 by default;
for $\tau = 1 \to \infty$ **do**
    Let $n = 2^\tau$;
    Let $m = n^c$;
    if $\mathit{SIZE}(L_m^{\mathit{PSPACE}}) \leq n^b$ **then**
        Set $y_n = 1$;
    else
        Let $\ell = \max\{\ell : \mathsf{SIZE}(L_\ell^{\mathsf{PSPACE}}) \leq n^b\}$;
        Set $y_{m+\ell} = 1$;

The Algorithm Satisfies the MA ∩ coMA Promise. We first show the algorithm satisfies the MA ∩ coMA promise (Definition 21). The intuition is that it only tries to guess-and-verify a circuit for L^{PSPACE} when it exists, and the properties of the instance checker (Definition 17) ensure that in this case the algorithm satisfies the MA ∩ coMA promise. Let $y = y_n$, there are three cases:

1. $y = 0$. In this case, the algorithm computes the all zero function, and clearly satisfies the MA ∩ coMA promise.
2. $y = 1$ and n is a power of 2. In this case, from Algorithm 2, we know that $\mathsf{SIZE}(L_m^{\mathsf{PSPACE}}) \leq n^b$ for $m = n^c$. Therefore, at least one guess of the circuit is the correct circuit for L_m^{PSPACE}, and on that guess, the algorithm outputs $L_n^{\mathsf{diag}}(x) = L_m^{\mathsf{PSPACE}}(z)$ with probability at least $2/3$, by the property of the instance checker (Definition 17).
   Again by the property of the instance checker, on all possible guesses, the algorithm outputs $1 - L_n^{\mathsf{diag}}(x) = 1 - L_m^{\mathsf{PSPACE}}(z)$ with probability at most $1/3$. Hence, the algorithm correctly computes L_n^{diag} on inputs of length n, with respect to Definition 21.
3. $y = 1$ and n is not a power of 2. In this case, from Algorithm 2, we know that $\mathsf{SIZE}(L_\ell^{\mathsf{PSPACE}}) \leq n_0^b$. Therefore, at least one guess of the circuit is the correct circuit for L_ℓ^{PSPACE}, and on that guess, the algorithm outputs $L_\ell^{\mathsf{PSPACE}}(z)$ ($z = z(x)$ is the first ℓ bits of x) with probability at least $2/3$, by the property of the instance checker (Definition 17).
   Again by the property of the instance checker, on all possible guesses, the algorithm outputs $1 - L_\ell^{\mathsf{PSPACE}}(z)$ with probability at most $1/3$. Hence, the algorithm correctly computes $L_\ell^{\mathsf{PSPACE}}(z(x))$ on inputs of length n, with respect to Definition 21.

The Algorithm Computes a Hard Language. Next we show that the algorithm indeed computes a hard language as stated. Let τ be a sufficiently large integer, $n = 2^\tau$, and $m = n^c$. According to Algorithm 2, there are two cases:

- $\mathsf{SIZE}(L_m^{\mathsf{PSPACE}}) \leq n^b$. In this case, Algorithm 2 sets $y_n = 1$. And by previous analyses, we know that L_n computes the hard language L_n^{diag}, and therefore $\mathsf{SIZE}(L_n) > n^k$.
- $\mathsf{SIZE}(L_m^{\mathsf{PSPACE}}) > n^b$. Let ℓ be the largest integer such that $\mathsf{SIZE}(L_\ell^{\mathsf{PSPACE}}) \leq n^b$. By Remark 18, we have $0 < \ell < m$.
  Note that $\mathsf{SIZE}(L_{\ell+1}^{\mathsf{PSPACE}}) \leq (\ell+1)^d \cdot \mathsf{SIZE}(L_\ell^{\mathsf{PSPACE}})$ for a universal constant d, because L^{PSPACE} is downward self-reducible. Therefore,

  $$\mathsf{SIZE}(L_\ell^{\mathsf{PSPACE}}) \geq \mathsf{SIZE}(L_{\ell+1}^{\mathsf{PSPACE}})/(\ell+1)^d \geq n^b/m^d \geq n^{b-c\cdot d}.$$

  Now, on inputs of length $m_1 = m + \ell$, we have $y_{m_1} = 1$ by Algorithm 2 (note that $m_1 \in (m, 2m)$ as $\ell \in (0, m)$). Therefore, L_{m_1} computes L_ℓ^{PSPACE}, and

  $$\mathsf{SIZE}(L_{m_1}) = \mathsf{SIZE}(L_\ell^{\mathsf{PSPACE}}) \geq n^{b-c\cdot d}.$$

  We set b such that $n^{b-cd} \geq (2m)^k \geq m_1^k$ (we can set $b = cd + 3 \cdot ck$), which completes the proof. ◀

A Fine-Grained Analogue of Schaefer's Theorem in P: Dichotomy of $\exists^k\forall$-Quantified First-Order Graph Properties

Karl Bringmann
Max Planck Institute for Informatics, Saarland Informatics Campus (SIC), Saarbrücken, Germany

Nick Fischer
Max Planck Institute for Informatics, Saarbrücken Graduate School of Computer Science, Saarbrücken, Germany

Marvin Künnemann
Max Planck Institute for Informatics, Saarland Informatics Campus (SIC), Saarbrücken, Germany

Abstract

An important class of problems in logics and database theory is given by fixing a first-order property ψ over a relational structure, and considering the model-checking problem for ψ. Recently, Gao, Impagliazzo, Kolokolova, and Williams (SODA 2017) identified this class as fundamental for the theory of fine-grained complexity in P, by showing that the *(Sparse) Orthogonal Vectors* problem is complete for this class under fine-grained reductions. This raises the question whether fine-grained complexity can yield a precise understanding of all first-order model-checking problems. Specifically, can we determine, for any fixed first-order property ψ, the exponent of the optimal running time $O(m^{c_\psi})$, where m denotes the number of tuples in the relational structure?

Towards answering this question, in this work we give a dichotomy for the class of $\exists^k\forall$-quantified graph properties. For every such property ψ, we either give a polynomial-time improvement over the baseline $O(m^k)$-time algorithm or show that it requires time $m^{k-o(1)}$ under the hypothesis that MAX-3-SAT has no $O((2-\varepsilon)^n)$-time algorithm. More precisely, we define a hardness parameter $h = H(\psi)$ such that ψ can be decided in time $O(m^{k-\varepsilon})$ if $h \leq 2$ and requires time $m^{k-o(1)}$ for $h \geq 3$ unless the h-uniform HyperClique hypothesis fails. This unveils a natural hardness hierarchy within first-order properties: for any $h \geq 3$, we show that there exists a $\exists^k\forall$-quantified graph property ψ with hardness $H(\psi) = h$ that is solvable in time $O(m^{k-\varepsilon})$ *if and only if* the h-uniform HyperClique hypothesis fails. Finally, we give more precise upper and lower bounds for an exemplary class of formulas with $k = 3$ and extend our classification to a counting dichotomy.

2012 ACM Subject Classification Theory of computation → Problems, reductions and completeness

Keywords and phrases Fine-grained Complexity, Hardness in P, Hyperclique Conjecture, Constrained Triangle Detection

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.31

1 Introduction

One of the most expressive problems in complexity theory is the model-checking problem for first-order definable properties over relational structures. Any such property can be written as a formula of the form

$$(Q_1 x_1)\ \ldots\ (Q_k x_k)\, \phi(x_1, \ldots, x_k),$$

where $Q_i \in \{\exists, \forall\}$, ϕ is an arbitrary Boolean formula defined over an arbitrary set of predicates and the relational structure is given by explicitly listing all tuples defining the predicates. This problem encompasses, e.g., the hugely diverse set of constraint satisfaction problems and

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka; Article No. 31; pp. 31:1–31:27

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

it is fundamental for database theory as *query evaluation* problem over relational structures. In this context, the relational structure is a relational database and the first-order definable property corresponds to queries to the database, see, e.g., [9] for an overview.

Given its expressiveness, it is no surprise that the complexity of this problem has been extensively studied under various angles: Among others, one distinguishes between the combined complexity (where both the first-order property and the relational structure are part of the input) and the data complexity (where the first-order property is fixed and the input only contains the relational structure) [54]. After classical works covered various aspects of these complexities (see, e.g., [31, 10, 30, 44]), later research turned towards parameterized analyses of such problems, see, e.g., [55, 37, 50] and the overview in [40, Section 4.3]. In this work, we pursue an even finer-grained complexity analysis of the data complexity of bounded-variable formulas, which capture a rich complexity landscape of low-degree polynomial-time problems [58, 41].

Consider the following examples for first-order properties, where the relational structure consists of the binary *edge relation* $E(x, x')$ over vertices in V, defining an (undirected) graph G:

- Triangle: $(\exists x_1 \in V)\,(\exists x_2 \in V)\,(\exists x_3 \in V)\,(E(x_1, x_2) \wedge E(x_2, x_3) \wedge E(x_3, x_1))$
  (determine whether G contains a triangle)
- TwinFree: $(\forall x_1 \in V)\,(\forall x_2 \in V)\,(\exists x_3 \in V)\,(\overline{E(x_1, x_2)} \vee (E(x_1, x_3) \not\leftrightarrow E(x_2, x_3)))$
  (determine whether G contains no adjacent vertices x_1, x_2 sharing the same neighborhood)

Such properties are called *graph properties*. Another example for a graph property (although we do not call the binary relation "edges" in this case) is a version of the Hitting Set problem, in which we are given an explicitly represented set family $\mathcal{S} = \{S_1, \ldots, S_n\}$ over some universe U. The explicit representation here is a list of tuples fulfilling the relation $s \in S_i, i \in \{1, \ldots, n\}$:

- HittingSet: $(\exists H \in \mathcal{S})\,(\forall S \in \mathcal{S})\,(\exists u \in U)\,(u \in H \wedge u \in S)$
  (determine whether there is some input set H that hits all (other) input sets)

A simple baseline algorithm solves the model-checking problem for any first-order formula in prenex normal form with $(k+1)$ quantifiers in time $O(m^k)$, where m denotes the size of the relational structure (i.e., the number of tuples satisfying the relations). This has recently been improved to $m^k/2^{\Omega(\sqrt{\log m})}$ [41]. However, we can surpass this bound significantly for specific formulas. In particular, TwinFree has a simple $O(m)$-time algorithm [42] and Triangle can be solved in time $O(m^{\frac{2\omega}{\omega+1}}) = O(m^{1.41})$ [12] (where $\omega \leq 2.373$ denotes the matrix multiplication exponent), while for HittingSet we do not know of any faster algorithm than the $m^2/2^{\Omega(\sqrt{\log m})}$-solution of Gao et al. [41] (in fact, this barrier has been used as a hardness assumption in its own right [6]). This raises the question: How can we determine the constant $c_\psi > 0$ in the optimal running time $m^{c_\psi \pm o(1)}$ for specific formulas ψ? In particular, when is a close-to-baseline time $m^{k \pm o(1)}$ the best possible?

1.1 Complete Problem: (Sparse) k-OV

Remarkable progress to this question has been made by Gao, Impagliazzo, Kolokolova, and Williams [41]. In the sense of admitting polynomial improvements over the $O(m^k)$-baseline, they identified the following problem as *complete for the class of model-checking problems for $(k+1)$-quantifier formulas*: (here, the input is a $(k+1)$-partite graph $G = (X_1 \uplus \cdots \uplus X_k \uplus Y, E)$)

- Sparse k-OV: $(\exists x_1 \in X_1) \ldots (\exists x_k \in X_k)\,(\forall y \in Y)\,(\overline{E(x_1, y)} \vee \ldots \vee \overline{E(x_k, y)})$

More precisely, Gao et al. show that Sparse k-OV has an $O(m^{k-\varepsilon})$-time algorithm for some constant $\varepsilon > 0$ if and only if for *all* $(k+1)$-quantifier formulas in prenex normal form, the model-checking problem can be solved in time $O(m^{k-\varepsilon'})$ for some constant $\varepsilon' > 0$. This identifies Sparse k-OV as one of the hardest problems in this class of problems.

Note that Sparse k-OV is a sparsely represented variant[1] of the following problem:

▶ **Problem 1** (k-OV)**.** Given k sets of 0-1 vectors $A_1, \ldots, A_k \subseteq \{0,1\}^d$, where $|A_i| = n$, determine whether there is a tuple $a_1 \in A_1, \ldots, a_k \in A_k$ such that $\prod_{i=1}^{k} a_i[\ell] = 0$ for all $\ell \in [d]$.

It is conjectured that for any constant k, k-OV cannot be solved in time $O(n^{k-\varepsilon}\operatorname{poly}(d))$ for any constant $\varepsilon > 0$, which is called the k-OV conjecture. Assuming this lower bound only for $k = 2$ is known simply as OV conjecture – this has been used as a hardness assumption for a number of conditional lower bounds in the quadratic-time regime [52, 18, 8, 13, 4, 20, 14, 6, 19, 21, 59, 2]. Likewise, the stronger k-OV conjecture has found uses to derive further polynomial-time lower bounds of different degree [51, 4, 1, 15, 48].

For almost a decade, the main support for the k-OV hypothesis was given by a reduction by Williams [57] using the so-called split-and-list technique to show that the Strong Exponential Time Hypothesis [45] implies the k-OV hypothesis. Only recently, Abboud et al. [5] show that the OV hypothesis is also implied by the weighted k-Clique hypothesis. Interestingly, the work by Gao et al. [41] gives additional evidence, as they show that existence of an $O(n^{k-\varepsilon}\operatorname{poly}(d))$-time algorithm for k-OV for constant $\varepsilon > 0$ is equivalent to the existence of an $O(m^{k-\varepsilon'})$-time model-checking algorithm for all $(k+1)$-quantifier formulas in prenex normal form for some constant $\varepsilon' > 0$. The study of first-order properties is thus tightly connected to the study of hardness and structure within P. In particular, our aim is to understand which properties make a first-order formula expressive enough to capture the hardest model-checking problems, i.e., Sparse k-OV, and which properties make a first-order formula easier to evaluate.

1.2 Classification à la Schaefer

Our questions ask for fine-grained analogues of classical results in computational complexity theory. Specifically, consider the case of Boolean constraint satisfaction problems (CSPs). As each Boolean CSP is in NP, Cook's theorem establishes that every CSP reduces to 3-SAT. Schaefer's Theorem [53] proves a dichotomy for reductions in the reverse direction: Assuming $\mathsf{P} \neq \mathsf{NP}$, every Boolean CSP either is polynomial-time solvable or requires superpolynomial time via a reduction from 3-SAT.

Translated to our setting, where first-order properties correspond to the class NP, Gao et al. give an analogue of Cook's Theorem: They give a fine-grained reduction from the model-checking problem of any $(k+1)$-quantifier first-order formula to Sparse k-OV. This raises the question:

Can we give a dichotomy analogous to Schaefer's classification, i.e., for each such formula either give an $O(m^{k-\varepsilon})$-time algorithm or show "Sparse k-OV-hardness"?

[1] To see the correspondence, let the vertex sets $X_i, i \in \{1, \ldots, k\}$ represent the vector sets A_i, let Y denote the dimensions $\{1, \ldots, d\}$ and let the binary relation $E(x_i, y)$ with $x_i \in X_i$ and $y \in Y$ hold if and only if $x_i[y] = 1$. We call this representation *sparse* as the relational structure only lists the 1-entries of the k-OV instance.

Our aim in this work is to initiate the investigation of such fine-grained classifications into *hardest* and *easier* first-order properties. Note that in the case of CSPs over finite domains, this line of research proved to be an effort requiring four decades of research in complexity, logic and algebra (see e.g., [34] for an early overview and the surveys [32, 25] for an introduction to the algebraic approach). In particular, after Schaefer's classification of the Boolean domain, Feder and Vardi raised the dichotomy conjecture that such a dichotomy also exists for CSPs over arbitrary finite domains [38, 39]. After a series of works developing an algebraic view on CSPs [47, 46, 26], and classifications of larger classes of CSPs (e.g., [43, 22, 23]), only very recently, Bulatov [24] and Zhuk [60] could finally resolve the conjecture.

Given the close connection to CSPs, we do not expect a full dichotomy for bounded-quantifier first-order properties to be within immediate reach of current techniques – thus, we focus on expressive fragments first. In particular, we focus on formulas with the quantifier structure $\exists^k\forall$, as it is the quantifier structure of the known complete problem Sparse k-OV (in fact, under a nondeterministic variant of SETH, $\exists^k\forall$ and the symmetric $\forall^k\exists$ are the only quantifier structure containing complete problems for first-order properties [28]). Note also that this quantifier structure is analogous to the quantifier structure for CSP solvability (existential quantifiers for variable assignments and a universal quantifier to check that all constraints are satisfied; this correspondence is best illustrated by Williams' split-and-list reduction from CNF satisfiability to OV [57]). We leave the remaining quantifier structures (analogous to the classification of quantified CSPs [53, 35, 34]) to be addressed in future work.

1.3 Further Related Work

Note that related work in database theory gives further flavors of fine-grained dichotomies for first-order properties: For the related setting of *query enumeration*, Bagan, Durand and Grandjean [16] classify each acyclic conjunctive query as either admitting constant-delay enumeration following linear-time precomputation or as hard under the assumption that Boolean matrix multiplication requires superquadratic time. This classification was recently extended to incorporate functional dependencies between attributes [27]. Further work gives fine-grained dichotomies under the OMv and OV hypotheses for dynamic databases [17].

1.4 Our Results

Our main result is a dichotomy for $\exists^k\forall$-quantified formulas over graphs under a plausible assumption about the complexity of MAX-3-SAT. Formally, $\exists^k\forall$-quantified first-order graph properties are formulas of the form

$$\psi = (\exists x_1 \in X_1) \ldots (\exists x_k \in X_k)\,(\forall y \in Y)\,\phi(x_1, \ldots, x_k, y), \tag{1}$$

where ϕ is an arbitrary Boolean formula defined over the atoms $E(v, v')$ with $v, v' \in \{x_1, \ldots, x_k, y\}$ and $v \neq v'$. Let $\mathrm{MC}(\psi)$ denote the corresponding model-checking problem: Check whether ψ holds for a given a $(k+1)$-partite[2] graph with vertex parts $X_1, \ldots, X_k, Y$.

▶ **Theorem 1** (Dichotomy). *For any $\exists^k\forall$-quantified graph property ψ, deciding* $\mathrm{MC}(\psi)$ *either requires time $m^{k-o(1)}$ under the assumption that* MAX-3-SAT *has no $O((2-\varepsilon)^n)$-time algorithm for any $\varepsilon > 0$, or we give an $O(m^{k-\varepsilon})$-time algorithm for some $\varepsilon > 0$.*

[2] A discussion on this assumption follows in Section 2.

In fact, we base our hardness results even on the weaker 3-uniform HyperClique assumption as introduced in [5, 49]. Formally, the h-uniform HyperClique hypothesis states for any parameter $h \geq 3$:

▶ **Hypothesis** (h-uniform HyperClique). *For no $\varepsilon > 0$ and $k \geq h+1$, there is an $O(n^{k-\varepsilon})$-time algorithm for detecting a k-clique in h-uniform hypergraphs.*

We defer a discussion of the plausibility of the MAX-3-SAT and h-uniform HyperClique hypotheses to Section 2.2 and the detailed treatment in [49, Section 7].

Beyond Theorem 1, we gain deeper insights into the complexity landscape of first-order graph properties. In particular, we expose a fine-grained hardness hierarchy purely depending on a hardness parameter $h = H(\psi)$ which we define below (illustrated in Figure 1): if $h = k$, then a lower bound of $m^{k-o(1)}$ can be derived from the k-OV conjecture, and thereby from SETH. On the other extreme, if $h \leq 2$, we give $O(m^{k-\varepsilon})$-time algorithms (for some $\varepsilon > 0$) – here, the difference between hardness 1 and 2 is precisely whether or not fast matrix multiplication techniques are likely to be necessary. For the remaining cases of $3 \leq h < k$, we can derive a lower bound of $m^{k-o(1)}$ under the 3-uniform HyperClique conjecture. In fact, we obtain increasing levels of hardness, as the lower bound for hardness-h formulas follows from the h-uniform HyperClique conjecture.

For the definition of our hardness parameter, it turns out that the decisive information is given by the atoms $E(x_i, y)$ for some existentially quantified variable x_i and the universally quantified y. Specifically, consider the formula ϕ_0 obtained by setting all atoms $E(x_i, x_j), 1 \leq i < j \leq k$ in ϕ to *false*. Observe that we can view ϕ_0 as a Boolean function $\{0,1\}^k \to \{0,1\}$ which maps the values $(E(x_1, y), \ldots, E(x_k, y))$ to a truth value. The hardness h of ψ is then given by the following hardness parameter $H(\phi_0)$. To state its definition, we need the following notation: For a propositional formula $f(z_1, \ldots, z_k)$ and an index set $I \subseteq [k]$, an *I-restriction of f* is a formula obtained from f after substituting all variables z_i, $i \in I$, by constant values from $\{0,1\}$.

▶ **Definition 2.** *We call a propositional formula $f(z_1, \ldots, z_k)$* h-hard, *$0 \leq h \leq k$, if, for any index set $I \in \binom{[k]}{k-h}$, there exists some I-restriction of f with exactly one falsifying assignment. Further define the* hardness $H(\psi)$ *as the maximum number h for which ψ is h-hard (for constant-valued f, we set $H(f) = 0$).*

Intuitively, $H(f)$ is the largest arity k such that whenever we fix an arbitrary subset of all but k variables, we can still obtain a "k-OV-like" function (a function with only a single falsifying assignment) as a restriction.

The following theorem is a fine-grained version of our dichotomy in Theorem 1.

▶ **Theorem 3** (Hardness levels). *For a first-order property ψ as in (1), let $\phi_0 : \{0,1\}^k \to \{0,1\}$ denote the formula obtained from ϕ by replacing all occurrences of $E(x_i, x_j)$ by false. We call $H(\psi) := H(\phi_0)$ the* hardness of ψ. *For $h = H(\psi)$, it holds that*

- *If $h \leq 1$, then $\mathrm{MC}(\psi)$ is decidable in time $O(m^{k-\varepsilon})$ for some $\varepsilon > 0$ combinatorially*[3].
- *If $h \leq 2 < k$, then $\mathrm{MC}(\psi)$ is decidable in time $O(m^{k-\varepsilon})$ for some $\varepsilon > 0$ using fast matrix multiplication. (Furthermore, $\mathrm{MC}(\psi)$ cannot be decided by a combinatorial $O(m^{k-\varepsilon})$-time algorithm unless the combinatorial k-Clique hypothesis*[4] *is false.)*

[3] Informally, by *combinatorial* algorithms we mean algorithms that do not rely on algebraic methods like fast matrix multiplication [3].

[4] The *combinatorial k-Clique hypothesis* as stated in, e.g., [3, 49] postulates that – even though an $O(n^{\frac{\omega}{3}k})$-time algorithm is known for k-Clique – no such polynomial improvement over the naïve $O(n^k)$ solution can be achieved by a *combinatorial* algorithm.

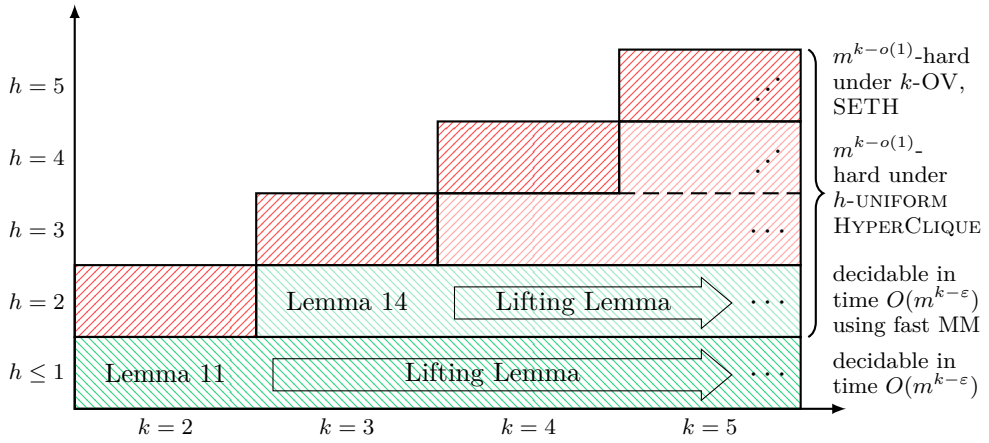


Figure 1 Visualizes the hardness of $\mathrm{MC}(\psi)$ for $\exists^k\forall$-quantified graph properties ψ of hardness $h = H(\psi)$. The green-hatched areas designate instances that allow polynomial improvements over the baseline, while the red regions turn out to be provably hard.

- *If $3 \leq h \leq k$, then $\mathrm{MC}(\psi)$ cannot be decided in time $O(m^{k-\varepsilon})$ for any $\varepsilon > 0$ unless the h-*UNIFORM HYPERCLIQUE *hypothesis fails.*
- *If $h = k$, then $\mathrm{MC}(\psi)$ cannot be decided in time $O(m^{k-\varepsilon})$ for any $\varepsilon > 0$ unless the k-*OV *hypothesis fails.*

Note that under the plausible assumption that h-UNIFORM HYPERCLIQUE gets strictly harder for increasing arity h, our classification exposes increasing levels of hardness within the first-order graph properties. This claim is substantiated by the following observation (whose proof is deferred to the full version of the paper).

▶ **Proposition 4.** *Let $h \geq 3$. There exist $k > h$, $\varepsilon > 0$, and an $\exists^k\forall$ graph property ψ of hardness h that can be decided in time $O(m^{k-\varepsilon})$ if and only if the h-*UNIFORM HYPERCLIQUE *hypothesis fails.*

To give a specific illustration of our results, consider the first-order property TWINFREE. By negating the formula, we obtain an equivalent graph property as in (1) where ϕ_0 is the constant false formula. As such, our classification yields that TWINFREE is decidable in time $O(m^{2-\varepsilon})$ (in fact, it is even decidable by an $O(m)$-time algorithm [42]).

Another interesting family of examples is the following k-Not-All-Equal Problem.

▶ **Example 5.** Let $\mathrm{NAE}(z_1, \ldots, z_k)$ be falsified only by the all-zero or all-one assignment. The k-Not-All-Equal (k-NAE) problem is given by the query

$$(\exists x_1 \in X_1) \ldots (\exists x_k \in X_k)\,(\forall y \in Y)\,\mathrm{NAE}(E(x_1, y), \ldots, E(x_k, y)).$$

It is easy to check that $H(k\text{-NAE}) = k - 1$. Thus, by Theorem 3,

- 2-NAE is decidable in time $O(m^{2-\varepsilon})$ for some $\varepsilon > 0$.
  (In fact, we give an $O(m)$-time algorithm.)
- 3-NAE is decidable in time $O(m^{3-\varepsilon})$ for some $\varepsilon > 0$ using fast matrix multiplication,
- k-NAE takes time $m^{k-o(1)}$ for any $k > 3$ unless the $(k-1)$-UNIFORM HYPERCLIQUE hypothesis fails.

Finally, we extend our results and discuss further directions in Section 7: In particular, we extend Theorem 3 to a counting dichotomy. Furthermore, we give tighter bounds on the running time exponent for properties that admit polynomial improvements over the baseline, using $k = 3$ as a case study.

1.5 Outline

After setting up notation and detailing the hardness assumptions used in this work in Section 2, we give a technical overview of our proof and introduce our main algorithmic tools in Section 3. Following the outline described in Section 3, we prove our main result in Sections 4, 5 and 6. Finally, we discuss our extensions and give an outlook for future work in Section 7. The proofs of our extensions are deferred to the full version of this paper.

2 Preliminaries

Let us clarify some notation first. For a non-negative integer k, let $[k] := \{1, \dots, k\}$. By $\uplus$ we denote the disjoint union of sets and for any set I, by $\binom{I}{k}$ we address the set of all k-element subsets of I. For a 0-1 vector x, we write $\bar{x}$ for the complement of x and $\|x\|$ to denote the Hamming weight (that is, the ℓ_1-norm) of x. Occasionally, we apply bit-wise binary operations to vectors understood as component-wise application. We further employ the Iverson bracket notation, that is, we write $[P]$ to denote the truth value of a proposition P. Let $f(z_1, \dots, z_k)$ be a propositional formula and let $I \subseteq [k]$. For any assignment $\alpha : I \to \{0, 1\}$, we write $f|_\alpha$ to denote the formula obtained from f after substituting z_i by α_i, for all $i \in I$. Finally, by $\omega \leq 2.373$ we denote the exponent of matrix multiplication.

2.1 Model-Checking

A *relational structure* consists of n objects and predicates of arbitrary arity relating these objects. These predicates are explicitly given as lists of records; let m denote the total number of such facts. Without loss of generality we assume $n \leq O(m)$ by ignoring objects not occurring in any relation. A *first-order property* is given by a quantified formula

$$(Q_1 x_1) \dots (Q_k x_k)\, \phi(x_1, \dots, x_k),$$

where each quantifier $Q_i \in \{\exists, \forall\}$ ranges over all objects of the relational structure. The proposition ϕ is allowed to contain Boolean connectives and its atoms are given by predicates relating the quantified objects. The problem $\mathrm{MC}(\psi)$ of checking whether a fixed first-order property ψ holds on a given sparse structure is called the *model-checking problem* (or *query evaluation problem*) for ψ.

In this paper, we consider a fragment that we call $\exists^k\forall$*-quantified graph properties*: Here the input is a $(k+1)$-partite graph $G = (X_1 \uplus \dots \uplus X_k \uplus Y, E)$ and the task is to model-check the fixed formula

$$\psi = (\exists x_1 \in X_1) \dots (\exists x_k \in X_k)\, (\forall y \in Y)\, \phi(x_1, \dots, x_k, y),$$

where ϕ is an arbitrary Boolean formula formed from a single edge predicate E of arity 2 (i.e., the atoms of ϕ are of the form $E(v, v')$ with $v, v' \in \{x_1, \dots, x_k, y\}$). We assume the edge predicate to be symmetric (i.e., G to be a symmetric graph). We adopt the convention that x_i (respectively y) ranges over X_i (respectively Y) and therefore omit to explicit state membership $x_i \in X_i$ when it is clear from the context. Borrowing the notation from the definition of Sparse OV, we let $d := |Y|$ denote the number of objects in the range of the universally quantified variable. Since $\mathrm{MC}(\psi)$ is solvable in time $O(m)$ for $k \leq 2$, we assume throughout the paper that $k \geq 2$.

Let us explicitly highlight a subtle point: An alternative natural formalization of graph properties would be to omit the assumption that the given graph is $(k+1)$-partite. This alters the flavor of model-checking problems slightly: While all of our upper bounds would also

hold for this alternative formulation (by replicating the vertex set), the assumption cannot be neglected for the lower bounds. In fact, there exist examples of first-order properties that we prove hard if each quantifier ranges over its own part, and which turn out as easy if instead each quantifier ranges over the whole vertex set.[5] We leave open an investigation of this alternative formulation for future work.

2.2 Hardness Assumptions

We briefly collect the hardness assumptions used in our classification result.

▶ **Hypothesis** (k-OV). *For no k and $\varepsilon > 0$, k-OV on n vectors with dimension d can be solved in time $O(n^{k-\varepsilon} \operatorname{poly} d)$.*

The fastest known algorithm solves k-OV in time $n^{2-\Omega(1/\log(d/\log n))}$ [7, 29]. On the hardness side, the k-OV hypothesis is implied by SETH [57], the weighted k-Clique hypothesis [5], and the hypothesis that not all $(k+1)$-quantifier first-order properties can be solved in time $O(m^{k-\varepsilon})$ [41]. We remark that we use the *moderate-dimensional* k-OV hypothesis here (in fact, SETH even implies a stronger version postulating hardness just above logarithmic dimension).

The most important hypothesis for this work concerns the HyperClique problem: Given an h-uniform hypergraph H, the h-uniform k-HyperClique problem asks to find vertices $v_1, \dots, v_k \in V(H)$ so that any size-h subset of $\{v_1, \dots, v_k\}$ is contained in $E(H)$. This gives rise to the following hypothesis for any $h \geq 3$.

▶ **Hypothesis** (h-uniform HyperClique). *For no $\varepsilon > 0$ and $k \geq h+1$, there is an $O(n^{k-\varepsilon})$-time algorithm for h-uniform k-HyperClique.*

The restriction $h \geq 3$ is indeed essential: The 2-uniform k-HyperClique problem – i.e., the k-Clique problem on ordinary graphs – is known to admit faster solutions. Lincoln et al. [49] provides a detailed analysis on why to believe that the $h \geq 3$ case should be significantly harder: As a main argument, any improvement over the $O(n^k)$-time k-Clique algorithm traces back to fast matrix multiplication, however, Strassen-like algebraic techniques can provably not be applied to the HyperClique setting [49]. Moreover, there is a reduction from MAX-3-SAT to h-uniform k-HyperClique ($h \geq 3$), showing that the h-uniform HyperClique conjecture is entailed by the following MAX-3-SAT hypothesis, which is the simplest justification for our hardness results.

Specifically, consider the MAX-3-SAT problem, which asks, given a 3-SAT instance, to find an assignment maximizing the number of satisfied clauses.

▶ **Hypothesis** (MAX-3-SAT). *For all $\varepsilon > 0$,* MAX-3-SAT *cannot be solved in time $O((2-\varepsilon)^n)$.*

The currently fastest known algorithm for MAX-3-SAT runs in time $2^{n-o(n)}$ [11]. This assumption implies both the 3-uniform k-HyperClique [49] and the OV [5] hypotheses, and thus provides the currently *easiest* barrier for algorithmic improvements upon formulas that we classify as hard.

---

[5] Consider the property $\psi = (\exists x_1 \in V) \dots (\exists x_k \in V)\,(\forall y \in V) \bigvee_{i=1}^{k} E(x_i, y)$, which is equivalent to the *k-Dominating Set* problem. It is easy to see that $\mathrm{MC}(\psi)$ can be decided in time $O(m^{k-1})$: For any solution $(x_1, \dots, x_k)$, there must exist one "heavy" vertex x_i dominating at least n/k vertices y. However, there can be at most $O(m/n)$ many such vertices x_i. It is feasible to explicitly enumerate all heavy vertices and solve the remaining k-quantifier problem in $O(n^{k-2}m)$ time using the baseline algorithm. The total running time is $O(m/n \cdot n^{k-2}m) = O(m^{k-1})$. However, if the quantifiers of ψ range over separate sets $X_1, \dots, X_k, Y$, then deciding $\mathrm{MC}(\psi)$ requires time $m^{k-o(1)}$ (Theorem 3).

3 Technical Overview

To prove our result, we introduce the following type of $\exists^k\forall$-quantified graph properties, in which we interpret the input graph $G = (V = (X_1 \uplus \cdots \uplus X_k \uplus Y), E)$ as sets of *vectors* $X_1, \ldots, X_k$, by setting the entry $x_i[y] \in \{0,1\}$ to be 1 if and only if the edge (x_i, y) is present in G, i.e, $x_i[y] = [E(x_i, y)]$ (for any $x_i \in X_i, i \in [k]$, and $y \in Y$). For any Boolean function $\phi : \{0,1\}^k \to \{0,1\}$, we define the corresponding *Vector Problem* $\mathrm{VP}(\phi)$

$$(\exists x_1 \in X_1) \ \ldots \ (\exists x_k \in X_k)\,(\forall y \in Y)\,\phi(x_1[y], \ldots, x_k[y]).$$

Intuitively, Vector Problems are a proper subclass of $\exists^k\forall$-quantified graph properties, since the latter additionally considers the edges $E(x_i, x_j)$. Note that Sparse k-OV coincides with $\mathrm{VP}(\phi)$ for $\phi(z_1, \ldots, z_k) = \bigvee_{i=1}^k \bar{z_i}$; in particular, this function only has a single falsifying assignment.

We prove our main dichotomy (Theorems 1 and 3) by first proving an analogous dichotomy for Vector Problems (see Theorem 6) and then showing an equivalence between Vector problems and general $\exists^k\forall$-quantified graph properties (see Theorem 7).

For the first step, we show that the complexity of a Vector Problem $\mathrm{VP}(\phi)$ is determined by the parameter $H(\phi)$ as defined in Definition 2:

▶ **Theorem 6.** *Let ϕ be a k-variable formula of hardness $h = H(\phi)$.*

- *If $h \leq 1$, then $\mathrm{VP}(\phi)$ is decidable in time $O(m^{k-1})$ combinatorially.*
- *If $h \leq 2 < k$, then $\mathrm{VP}(\phi)$ is decidable in time $O(m^{k-\varepsilon})$ for some $\varepsilon > 0$ using fast matrix multiplication. (Furthermore, $\mathrm{VP}(\phi)$ cannot be decided by a combinatorial $O(m^{k-\varepsilon})$-time algorithm unless the combinatorial k-clique hypothesis is false.)*
- *If $3 \leq h \leq k$, then $\mathrm{VP}(\phi)$ cannot be solved in time $O(m^{k-\varepsilon})$ for any $\varepsilon > 0$ unless the h-uniform HyperClique hypothesis fails.*
- *If $h = k$, then $\mathrm{VP}(\phi)$ cannot be solved in time $O(m^{k-\varepsilon})$ for any $\varepsilon > 0$ unless the k-OV hypothesis fails.*

This result is established as follows (see Figure 1 for an illustration). On the algorithmic side, we show that:

1. For any 2-variable formula ϕ of hardness 1, $\mathrm{VP}(\phi)$ can be solved in time $O(m)$ (Section 4.1).
2. For any 3-variable formula ϕ of hardness 2, $\mathrm{VP}(\phi)$ can be solved in time $O(m^{3-\varepsilon})$ for some $\varepsilon > 0$ (Section 4.2). This is our technically most demanding contribution, for which we introduce a framework of *Constrained Triangle* problems solvable in subcubic time (Section 3.1).
3. These algorithms can be lifted to higher number of variables k (Section 4.3). The idea is to brute force over all but 2 or 3 variables and to apply the algorithms above. However, these algorithms are not directly applicable, since the brute-forcing step does not reduce to solving a *single* $\mathrm{VP}(\phi)$ instance, but to solving a *mixture* of $\mathrm{VP}(\phi_i)$ instances for a constant number of formulas $\phi_1, \ldots, \phi_\ell$. To overcome this issue, we consider *Hybrid Vector Problems* $\mathrm{VP}(\Phi)$ for a set of formulas Φ and show that our algorithms from above apply whenever $\max\{H(\phi) \mid \phi \in \Phi\} \leq 1$ or 2, respectively. This extension to Hybrid Vector Problems is made possible by the generality of our Constrained Triangle framework that allows for a surprisingly simple combination of constraints for different formulas.

On the hardness side, for any k-variable hardness-h formula ϕ, we give a fine-grained reduction from finding a k'-clique in h-uniform hypergraphs to the model-checking problem for ϕ, where k' is a sufficiently large constant. Intuitively (and somewhat oversimplified), the variables $x_1, \ldots, x_k$ choose a k-clique in a k-partite hypergraph and the $(\forall y)\,\phi(x_1[y], \ldots, x_k[y])$-part verifies that $(x_1, \ldots, x_k)$ indeed forms a clique. To this end, y ranges over the non-edges

of the hypergraph and verifies that the vertices $v_1, \dots, v_h$ of such a non-edge are not included in $x_1, \dots, x_k$. Specifically, let J denote the parts containing $v_1, \dots, v_k$. Then by our hardness definition, we find a way to assign values to $x_i[y]$ for all $x_i \in X_i$ with $i \in [k] \setminus J$ such that we can exclude exactly the vertices $v_1, \dots, v_h$ by finding suitable values for $x_i[y]$ for all $i \in X_i$ with $i \in J$. The proof is given in Section 5.

In the second step, we extend our classification from Vector Problems to all $\exists^k\forall$ graph properties by the following equivalence.

▶ **Theorem 7.** *Let $\psi = (\exists x_1) \dots (\exists x_k)\,(\forall y)\,\phi(x_1, \dots, x_k, y)$ be an $\exists^k\forall$ graph property and let ϕ_0 denote the formula ϕ after substituting each predicate $E(x_i, x_j)$ by false.*

- *If $\mathrm{MC}(\psi)$ is decidable in time $T(m)$, then $\mathrm{VP}(\phi_0)$ is decidable in time $O(T(m))$.*
- *If $\mathrm{VP}(\phi_0)$ is decidable in time $O(m^{k-\varepsilon})$ for some $\varepsilon > 0$, then $\mathrm{MC}(\psi)$ is decidable in time $O(m^{k-\varepsilon'})$ for some $\varepsilon' > 0$.*

For an intuition for the proof of the non-trivial direction from general properties to Vector Problems, let us call $(x_1, \dots, x_k)$ a *solution* if $(\forall y \in Y)\,\phi(x_1, \dots, x_k, y)$ holds. We reduce the problem of detecting a solution $(x_1, \dots, x_k)$ such that for some $1 \leq i < j \leq k$, the edge $E(x_i, x_j)$ is present to the problem of counting triangles in a (sparse) graph. The remaining solutions *almost* correspond to solutions of the Vector Problem ϕ_0 – however, we need to additionally ensure that *no* edge $E(x_i, x_j)$ is present in such a solution. We overcome this technical issue using a counting argument that there are few solutions with at least one edge $E(x_i, x_j)$ present.

Theorems 1 and 3 then follow by combining Theorem 6 and 7.

3.1 Constrained Triangles Framework

Let us detail our main algorithmic tool in advance. We develop a convenient framework to detect triangles subject to an arbitrary combination of some well-behaved constraints. We achieve subcubic-time algorithms by employing fast matrix multiplication in combination with a careful constraint-specific analysis.

The problem of detecting three pairwise adjacent vertices in a tripartite graph $G = (V_1 \uplus V_2 \uplus V_3, E)$, is referred to as Triangle. It serves us as a combinatorial intermediate problem that immediately benefits from the significantly improved running time of fast matrix multiplication over the $O(n^3)$-time naïve approach. However, Triangle is of relatively little expressiveness as the only way to encode information into a Triangle instance is to customize the edge set E. We therefore strengthen Triangle by allowing certain *constraints* to further restrict the set of feasible solutions (v_1, v_2, v_3). Specifically, we make use of two types of constraints:

Sum: For edge weights $w : E \to \mathbb{Z}$, where $\sum_{e \in E} |w(e)| \leq O(n^2)$, and a target $t \in \mathbb{Z}$, we require that $w(v_1, v_2) + w(v_2, v_3) + w(v_3, v_1) = t$.

Equal: For edge weights $w : E \to \mathbb{Z}$, we require that $w(v_1, v_2) = w(v_1, v_3)$.

This prepares us to introduce *Constrained Triangle problems* in an inductive fashion: As the base case, Triangle is viewed as a Constrained Triangle problem. In addition, for any Constrained Triangle problem Δ and any constraint $C \in \{\text{Sum}, \text{Equal}\}$, by $\Delta[C]$ we understand the Constrained Triangle problem which is – on top of all constraints restricting Δ – constrained by C. We remark that each constraint features its own weight function (which is given as part of the input), so in particular an instance of the Constrained Triangle problem $\text{Triangle}[C_1] \cdots [C_r]$ is equipped with r weight functions corresponding to the respective Sum and Equal constraints C_i.

Arguably, Constrained Triangle problems seem to be a convenient "interface" to the algorithmic power of fast matrix multiplication, as we can specifically tailor constraints in the desired manner. Indeed, even subject to any constant number of Sum and Equal constraints, finding triangles remains subcubic.

▶ **Lemma 8.** *Let Δ be a Constrained Triangle problem. Then Δ can be decided in time $O(n^{3-\varepsilon})$ for some $\varepsilon > 0$.*

The proof of Lemma 8 is by induction on the structure of Δ; we consider the three possible cases below in Fact 1, Lemma 9 and Lemma 10. A crucial observation which we will exploit often, is that for any Constrained Triangle problem Δ, given (v_1, v_2, v_3), we can test in constant time whether (v_1, v_2, v_3) is a solution of Δ.

By applying fast (Boolean) matrix multiplication, it is well-known that Triangle is decidable in subcubic time:

▶ **Fact 1.** Triangle *is decidable in time $O(n^\omega)$.*

This settles the induction base. In the following two lemmas, we assume efficient algorithms for Δ and aim to find algorithms for Δ[Sum] and Δ[Equal], respectively.

We focus on Δ[Sum] first. Note that the restriction $\sum_{e\in E} |w(e)| \leq O(n^2)$ is indeed necessary: For unbounded weights, the problem of finding an exact-weight triangle is not known and in fact conjectured not to be decidable significantly faster than $O(n^3)$ [56]. Nevertheless, under this condition we achieve an efficient Δ[Sum] algorithm:

▶ **Lemma 9.** *If Δ is decidable in time $O(n^{3-\varepsilon})$ for some $\varepsilon > 0$, then Δ[Sum] is decidable in time $O(n^{3-\varepsilon'})$ for some $\varepsilon' > 0$.*

Proof. We call an edge *large-weight* if its weight exceeds n^δ in absolute value (where δ is yet to be fixed) and *small-weight* otherwise. Our first step is to eliminate all large-weight edges. By assumption, since $\sum_{e\in E} |w(e)| \leq O(n^2)$, there can be at most $O(n^2/n^\delta) = O(n^{2-\delta})$ many such edges. Thus, it is feasible to enumerate all large-weight edges (x_i, x_j) and all vertices x_ℓ in the remaining part X_ℓ, for all distinct i, j, ℓ. For each triple considered in that way, we explicitly check that

- (x_i, x_j, x_ℓ) forms a triangle, and
- (x_i, x_j, x_ℓ) satisfies all constraints of Δ, and
- $w(x_i, x_j) + w(x_j, x_\ell) + w(x_\ell, x_i) = t$.

Since all these tests run in constant time, this whole step takes time $O(n^{2-\delta} \cdot n) = O(n^{3-\delta})$. We accept if a solution was found, and otherwise continue by safely removing all large-weight edges from the graph.

So from now on, we can assume that all remaining edges are small-weight. For any combination of weights $w_{12}, w_{23}, w_{31} \in \{-n^\delta, \ldots, n^\delta\}$ summing exactly to t, we create a Δ instance that only includes edges (v_i, v_j) of the weight $w(v_i, v_j) = w_{ij}$. We proceed to solve all these instances and report if a solution was found. By construction, any solution to an instance created in that way satisfies all constraints of Δ and the additional Sum constraint.

Solving a single Δ subinstance takes time $O(n^{3-\varepsilon})$ by assumption. There are $O(n^{2\delta})$ many combinations of weights $w_{12}, w_{23}, w_{31} \in \{-n^\delta, \ldots, n^\delta\}$ with $w_{12} + w_{23} + w_{31} = t$, so we need time $O(n^{2\delta} \cdot n^{3-\varepsilon}) = O(n^{3+2\delta-\varepsilon})$ to solve all instances. By setting $\delta := \frac{\varepsilon}{3}$, the claim follows. ◀

▶ **Lemma 10.** *If Δ is decidable in time $O(n^{3-\varepsilon})$ for some $\varepsilon > 0$, then Δ[Equal] is decidable in time $O(n^{3-\varepsilon'})$ for some $\varepsilon' > 0$.*

Proof. For any vertex $v_1 \in V_1$, we define $\deg_i(v_1)$ as the number of edges of weight i incident to v_1. First, we enumerate all edges (v_1, v_3), and if $\deg_{w(v_1,v_3)}(v_1) \leq n^{1-\delta}$ (where δ is a parameter to be chosen later), then enumerate all edges (v_1, v_2) of the same weight $w(v_1, v_3)$. When finding a triple (v_1, v_2, v_3) forming a triangle and fulfilling all constrained imposed by Δ, we accept. Otherwise, we can safely remove all edges (v_1, v_3) where $\deg_{w(v_1,v_3)}(v_1) \leq n^{1-\delta}$. We can similarly remove all edges (v_1, v_2) with $\deg_{w(v_1,v_2)}(v_1) \leq n^{1-\delta}$. So we can assume that for all weights i, and all vertices v_1, $\deg_i(v_1)$ is either 0 or at least $n^{1-\delta}$. This step takes time $O(n^2 \cdot n^{1-\delta}) = O(n^{3-\delta})$.

Since each vertex v_1 is incident to at most n edges, each v_1 can be incident to edges of at most $n/n^{1-\delta} = n^\delta$ different weights; we denote these weights by $w_1^{v_1}, \ldots, w_{n^\delta}^{v_1}$ in an arbitrary order. Our strategy is as follows: We create n^δ many Δ instances, where the i-th instance contains, for each vertex v_1, only those edges incident to v_1 which are of weight $w_i^{v_1}$. Notice that in each instance, all edges incident to one fixed v_1 are of the same weight, even though edges incident to different v_1's are in general of different weights. In this way, we can now simultaneously search for all triangles (v_1, v_2, v_3) satisfying Δ's constraints and satisfying $w(v_1, v_2) = w(v_1, v_3) = w_i^{v_1}$. Solving all instances takes time $O(n^\delta \cdot n^{3-\varepsilon}) = O(n^{3+\delta-\varepsilon})$, under the assumption that Δ can be solved in time $O(n^{3-\varepsilon})$.

In total, the running time is bounded by $O(n^{3-\delta} + n^{3+\delta-\varepsilon})$, which is subcubic, namely $O(n^{3-\frac{\varepsilon}{2}})$, for $\delta := \frac{\varepsilon}{2}$. ◀

4 Algorithmic Results

In this section, we show the algorithmic part of Theorem 6. In particular, we show that for an k-variable hardness-h formula ϕ the Vector Problem $\mathrm{VP}(\phi)$ is easy if $k = 2$ and $h \leq 1$ (Section 4.1), or if $k = 3$ and $h \leq 2$ (Section 4.2). In fact, for both scenarios, we demonstrate how to solve the following more general version of Vector Problems that discriminates among dimensions in such a way that we can assert different formulas ϕ for different dimensions.

For a set of k-variable formulas Φ, and given a sparse structure over the vertex set $X_1 \uplus \ldots \uplus X_k \uplus Y$, where each dimension $y \in Y$ is associated to a formula $\phi_y \in \Phi$, the *Hybrid Vector Problem* $\mathrm{VP}(\Phi)$ is to check

$$(\exists x_1 \in X_1) \ldots (\exists x_k \in X_k)\, (\forall y \in Y)\, \phi_y(x_1[y], \ldots, x_k[y]).$$

Our hardness notion is generalized to sets of formulas Φ by $H(\Phi) := \max\{H(\phi) : \phi \in \Phi\}$. (Observe that we could equivalently define Hybrid Vector Problems as Vector Problems tolerating a multi-valued unary predicate on Y.)

4.1 Algorithms for $k = 2$

We first introduce a useful tool: partition refinement [42]. There exists a (simple) data structure which, once initialized with a universe U, explicitly maintains a partition $\biguplus_P P$ of U. It supports an operation $\textsc{Refine}(S)$ that splits each part P into $P \cap S$ and $P \setminus S$ and runs in time $O(|S|)$. In addition, we can always query, for each universe element $u \in U$, its current part P (with $u \in P$) and iterate over all elements in P in time $O(|P|)$.

▶ **Lemma 11.** *Let Φ be a set of 2-variable formulas of hardness $H(\Phi) \leq 1$. Then $\mathrm{VP}(\Phi)$ is decidable in time $O(m)$.*

Proof. Let $\phi \in \Phi$. We first prove that $\mathrm{VP}(\phi)$ is linear-time decidable, and later show the same for $\mathrm{VP}(\Phi)$. Since we consider $k = 2$ variables, we have a function $\phi\colon \{0,1\}^2 \to \{0,1\}$. We exhaustively discriminate all possible shapes of ϕ:

Case 0: ϕ has 0 or 4 satisfying assignments. Then $\mathrm{VP}(\phi)$ is constantly rejecting or accepting and trivially decidable.

Case 1: ϕ has 1 satisfying assignment. Then ϕ is of the shape $\phi(z_1, z_2) = \phi_1(z_1) \wedge \phi_2(z_2)$. Deciding $\mathrm{VP}(\phi)$ is equivalent to asserting the satisfiability of $(\exists x_1)\,(\forall y)\,\phi_1(x_1[y])$ and $(\exists x_2)\,(\forall y)\,\phi_2(x_2[y])$ separately, which we do by removing all vectors x_1 and x_2 violating $(\forall y)\,\phi_1(x_1[y])$ and $(\forall y)\,\phi_2(x_2[y])$, respectively. Focus on the former problem: We first precompute all values $\|x_1\|$ by iterating over the edge set once. Then we exclude all vectors x_1 with $\|x_1\| < d$ in case that $\phi_1(z_1) = z_1$ or $\|x_1\| > 0$ in case that $\phi_1(z_1) = \bar{z}_1$. In the remaining cases $\phi_1(z_1) = \mathit{true}$ and $\phi_1(z_1) = \mathit{false}$ we keep all or no vectors x_1, respectively.

Case 2: ϕ has 2 satisfying assignments.

Case 2a: $\phi(z_1, z_2) = (z_1 \leftrightarrow z_2)$. Our algorithm relies on the partition refinement technique: Initialize the partition refinement structure with $U = X_1 \uplus X_2$. Then, for each $y \in Y$, invoke $\textsc{Refine}(N(y))$, where $N(y) \subseteq X_1 \uplus X_2$ denotes the neighborhood of y. We claim two vectors x_1 and x_2 lie in the same partition if and only if $(\forall y)\, x_1[y] \leftrightarrow x_2[y]$. Indeed, assume that x_1 and x_2 are equal (as vectors). Then for any dimension $y \in Y$, we have $x_1 \in N(y)$ if and only if $x_2 \in N(y)$. Hence, there exists no entry $y \in Y$ separating x_1 from x_2. On the other hand, if $x_1[y] \leftrightarrow x_2[y]$ does not hold for some $y \in Y$, then $x_1 \in N(y)$ but $x_2 \notin N(y)$ (or vice versa), so $N(y)$ splits any part containing both x_1 and x_2. This approach takes time $\sum_{y \in Y} O(|N(y)|) = O(m)$ for the refinement steps and time $\sum_P O(|P|) \leq O(n)$ (where P ranges over all parts) to ultimately check whether some pair (x_1, x_2) was not separated.

Case 2b: $\phi(z_1, z_2) = z_1 \oplus z_2$. Our goal is to reduce to the Case 2a by transforming all vectors x_1, x_2 into x_1', x_2' in such a way that we have $x_1[y] \oplus x_2[y]$ if and only if $x_1'[y] \leftrightarrow x_2'[y]$, for all $y \in Y$. The naive way to achieve this is to negate all vectors $x_2 \in X_2$; however, we then potentially increase the total number of 1-entries inordinately, so that we do not obtain an $O(m)$-time algorithm. We circumvent this issue as follows. Let us call a vector x *heavy* if $\|x\| > d/2$, and *light* otherwise. Observe that in any solution (x_1, x_2), we must have that precisely one of x_1 and x_2 is heavy (assuming without loss of generality that d is odd). Now, assign $x_i' := x_i$ if x_i is light and $x_i' := \bar{x}_i$ otherwise. Then, for any solution (x_1, x_2), where, say, x_1 is heavy, $(x_1', x_2') = (\bar{x}_1, x_2)$ is a solution of $\mathrm{VP}(x_1' \leftrightarrow x_2')$. The other direction is not immediate as there could exist light vectors x_1, x_2 with $x_1' = x_1 = x_2 = x_2'$. To avoid these false positives, we introduce a fresh dimension y with $x_1'[y] = [x_1 \text{ is heavy}]$ and $x_2'[y] = [x_2 \text{ is light}]$, for all vectors x_1, x_2. In doing so, we achieve that for any solution (x_1', x_2') exactly one of the vectors x_1, x_2 is heavy.

It remains to bound the running time of the complementations. Since there exist at most $O(m/d)$ heavy vectors, and complementing a single vector takes time $O(d)$, this step takes time $O(m/d \cdot d) = O(m)$. Furthermore, notice that m cannot increase by replacing heavy vectors with light ones.

Case 3: ϕ has 3 satisfying assignments. Then ϕ would have exactly one falsifying assignment, so it would have hardness 2. Since ϕ has hardness at most 1 by assumption, this case cannot occur.

To arrive at an algorithm solving the hybrid problem, we take a close look at the preceding arguments: Either $\mathrm{VP}(\phi)$ reduces to an instance of $\mathrm{VP}(z_1 \leftrightarrow z_2)$ over the original vertex set (Case 2) or we remove certain vertices and accept if afterwards both parts X_1 and X_2 are still non-empty (Cases 0 and 1). In the same way, we can solve the Hybrid Vector Problem $\mathrm{VP}(\Phi)$: For all formulas $\phi \in \Phi$ falling into the latter category, we identify and remove all bad vertices x_1 and x_2. Then, for all remaining $\phi \in \Phi$, we invoke the reduction to $\mathrm{VP}(z_1 \leftrightarrow z_2)$ and concatenate all vectors corresponding to the same vertex x_i. Finally, it remains to solve the combined $\mathrm{VP}(z_1 \leftrightarrow z_2)$ instance as shown in Case 2a. ◀

4.2 Algorithms for $k = 3$

Next, we will show that, for $k = 3$, any Vector Problem of hardness at most 2 reduces to a Constrained Triangle problem Δ. The reduction is always of the following form: Given a VP(ϕ) instance G with vertices $V(G) = X_1 \uplus X_2 \uplus X_3 \uplus Y$, the corresponding Constrained Triangle instance is a graph G' over the vertex set $V(G') = X_1 \uplus X_2 \uplus X_3$. Moreover, we wish to satisfy that, for all (x_1, x_2, x_3):

$$(\forall y)\, \phi(x_1[y], x_2[y], x_3[y]) \iff (x_1, x_2, x_3) \text{ is a solution of } \Delta \text{ on } G'.$$

To this end, the following sections describe how to encode ϕ by Equal and Sum constraints step-by-step. More precisely, each constraint "covers" a pair of falsifying assignments of ϕ. Let $(\alpha^1, \beta^1), \ldots, (\alpha^r, \beta^r)$ be pairs of falsifying assignments of ϕ containing each falsifying assignment at least once. Then we reduce VP(ϕ) to a Constrained Triangle problem $\Delta = \textsc{Triangle}[C_1] \cdots [C_r]$ with $C_i \in \{\textsc{Equal}, \textsc{Sum}\}$ such that for all i and for all (x_1, x_2, x_3):

$$(\forall y)\, (x_1[y], x_2[y], x_3[y]) \notin \{\alpha^i, \beta^i\} \iff (x_1, x_2, x_3) \text{ satisfies the constraint } C_i.$$

Note that we indeed need the restriction $H(\phi) \leq 2$, since we can only cover the falsifying assignments of ϕ by pairs (α^i, β^i) if ϕ does not have exactly one falsifying assignment.

Recall that in Section 3.1, we measured the complexity of Constrained Triangles in terms of the number of vertices n. By viewing G' as a graph of m vertices (by potentially adding isolated nodes), we obtain algorithms whose running time solely depends on m. Furthermore, in the special case of Sum constraints, we thereby allow a total edge weight of up to $O(m^2)$ instead of $O(n^2)$.

4.2.1 Equal Constraints

We start by covering falsifying assignments α, β of Hamming distance 2 using Equal constraints.

▶ **Lemma 12.** *Let $\alpha, \beta \in \{0,1\}^3$ be of Hamming distance 2. In time $\tilde{O}(m^2)$, we can determine the edge weights of an* Equal *constraint C such that $(\forall y)\, (x_1[y], x_2[y], x_3[y]) \notin \{\alpha, \beta\}$ holds if and only if (x_1, x_2, x_3) satisfies C.*

Proof. Let $\alpha = (\alpha_1, \alpha_2, \alpha_3)$ and $\beta = (\beta_1, \beta_2, \beta_3)$. Without loss of generality, we assume that α equals β exactly in the first position. An Equal constraint is employed as follows by choosing each weight to be a dimension-d bit-vector (for the moment). Let $w(x_1, x_2)[y] := 1$ if and only if $(x_1[y], x_2[y]) = (\alpha_1, \alpha_2)$ and analogously let $w(x_1, x_3)[y] := 1$ if and only if $(x_1[y], x_3[y]) = (\beta_1, \beta_3)$.

Let y be arbitrary. It is easy to check that we have $w(x_1, x_2)[y] \neq w(x_1, x_3)[y]$ if and only if $(x_1[y], x_2[y], x_3[y]) \in \{\alpha, \beta\}$. Indeed, suppose that $1 = w(x_1, x_2)[y] \neq w(x_1, x_3)[y] = 0$. By our choice of $w(x_1, x_2)$, we must have $x_1[y] = \alpha_1$ and $x_2[y] = \alpha_2$. But $w(x_1, x_3)[y] = 0$, hence $x_3[y] = \bar{\beta}_3 = \alpha_3$. In summary: $(x_1[y], x_2[y], x_3[y]) = \alpha$. Symmetrically, $0 = w(x_1, x_2)[y] \neq w(x_1, x_3)[y] = 1$ entails $(x_1[y], x_2[y], x_3[y]) = \beta$. The converse direction is immediate.

However, computing the vectors $w(x_1, x_i)$ explicitly would take time $O(m^3)$, since there are $O(m^2)$ weights and each vector is of dimension $d = O(m)$. We therefore employ the following succinct computation and compression:

1. Compute all weights $w(x_1, x_i)$, $i = 2, 3$, stored in *run-length encoding*. That is, for each weight $w(x_1, x_i)$ we store a sequence of numbers $\mathrm{RLE}(w(x_1, x_i)) := \ell_1 \ell_2 \cdots \ell_r$ indicating the positions of 0-1 alternations in $w(x_1, x_i)$, i.e., $w(x_1, x_i) = 0^{\ell_1}\, 1^{\ell_2}\, \cdots\, 0^{\ell_{r-1}}\, 1^{\ell_r}$ written as a bit-string.

2. Interpret the run-length encoded weight vectors as strings, sort these strings to combine equal weight vectors, and replace every weight vector by its rank in the resulting sorted sequence. This replaces each weight vector by a number in $\{0, \dots, O(m^2)\}$ so that two edges share the same label if and only if their weights are equal.

To implement the first step, we identify Y with $[d]$ in an arbitrary order. Fix some $i = 2, 3$ and some edge (x_1, x_i) and let a 0-1 alternation occur at y, i.e. $w(x_1, x_i)[y] \neq w(x_1, x_i)[y+1]$. We observe that, for any choice of α, β, at least one of the entries $x_1[y]$, $x_i[y]$, $x_1[y+1]$ or $x_i[y+1]$ is 1. Thus, it is feasible to explicitly consider all "event points" y with $x_1[y] = 1$ or $x_i[y] = 1$ in increasing order. Whenever an alternation is detected at $w(x_1, x_i)[y-1]$ or $w(x_1, x_i)[y]$ we appropriately append the run-length encoding of $w(x_1, x_i)$. For a fixed pair (x_1, x_i), this approach takes time $O(\|x_1\| + \|x_i\|)$. In total, we obtain time $\sum_{x_1,x_i} O(\|x_1\| + \|x_i\|) = O(n^2+m^2) = O(m^2)$. By the same argument, it follows that the total length of all run-length encoding $\sum_{x_1,x_i} |\mathrm{RLE}(w(x_1, x_i))|$ is bounded by $O(m^2)$.

In spirit, the proof ends here. However, in requiring the edge labels of Equal constraints to be integers instead of arbitrary objects (here, vectors in run-length encoding), we did not have to worry about the bit-size of the edge weights in Section 3.1. So our second step is to associate all vectors $w(x_1, x_i)$ with integers in $\{0, \dots, O(m^2)\}$. We interpret each run-length encoded weight $\mathrm{RLE}(w(x_1, x_i)) := \ell_1\ell_2 \cdots \ell_r$ as a string of length r over alphabet $\Sigma = [d]$. We sort these strings, which leaves all equal weights as contiguous intervals in the sorted sequence. Finally, we replace each weight by its rank in this sorted sequence (i.e., by the number of distinct weights preceding it in the sorted sequence). It is well-known that M strings of total length N over an alphabet of size $\mathrm{poly}(N)$ can be sorted in time $\tilde{O}(N)$ using tries. This yields time $\tilde{O}(m^2)$ in our application. ◀

4.2.2 Sum Constraints

It remains to cover falsifying assignments α, β of odd Hamming distance, for which we will use Sum constraints. We start with some useful observations.

We aim to check whether some vectors $(x_1, \dots, x_k)$ satisfy $(\forall y)\, \phi(x_1[y], \dots, x_k[y])$. Say ϕ is falsified only by the all-ones input. Then, clearly, it is sufficient to check that there exists no vertex y connected to all vertices $x_1, \dots, x_k$ – we call such a configuration a $[k]$-*star*. More generally, by an I-*star*, $I \subseteq [k]$, we understand a subgraph centered at a vertex $y \in Y$ such that all edges (x_i, y), $i \in I$, are present. Notice that, for any vectors $(x_1, \dots, x_k)$, $\| \bigwedge_{i\in I} x_i\|$ exactly counts the number of I-stars. So, in the above example of ϕ, we can decide whether $(\forall y)\, (x_1[y], \dots, x_k[y])$ by checking whether the number of $[k]$-stars $\| \bigwedge_{i=1}^{k} x_i\|$ equals zero.

But what about other formulas ϕ? For each falsifying assignment $\alpha \in \{0, 1\}^k$, the obvious generalization is to require $\| \bigwedge_{i=1}^{k} x_i^{\alpha_i}\| = 0$; here, and for the remainder of this section, we write $x^1 := x$ and $x^0 := \bar{x}$. The following observations suggest how to transform an arbitrary expression of this form into a linear combination of terms $\| \bigwedge_{i\in I} x_i\|$ without complemented occurrences:

▶ **Observation 1.** *For all vectors x, x', $\|x \wedge \bar{x}'\| = \|x\| - \|x \wedge x'\|$.*

By induction, we obtain the following generalization:

▶ **Observation 2.** *For all vectors $x, x_1, \dots, x_k$, $\|x \wedge \bigwedge_{i=1}^{k} \bar{x}_i\| = \sum_{I\subseteq[k]} (-1)^{|I|} \|x \wedge \bigwedge_{i\in I} x_i\|$.*

Thus, if we could precompute the number of I-stars $\| \bigwedge_{i\in I} x_i\|$, then we could efficiently test whether $(\forall y)\, \phi(x_1[y], \dots, x_k[y])$ holds:

▶ **Observation 3.** *Given vertices $(x_1, \dots, x_k)$ and the number of I-stars $\| \bigwedge_{i \in I} x_i \|$ for all $I \subseteq [k]$, for any formula ϕ we can decide in constant time whether $(\forall y)\, \phi(x_1[y], \dots, x_k[y])$.*

As an example, consider the 3-Not-All-Equal problem VP(NAE), where $\mathrm{NAE}(x_1, x_2, x_3)$ has two falsifying assignments: 111 and 000. Equivalently, we could require that the triple (x_1, x_2, x_3) satisfies $\|x_1 \wedge x_2 \wedge x_3\| = 0$ and $\|\bar{x}_1 \wedge \bar{x}_2 \wedge \bar{x}_3\| = 0$. By Observation 2, the second equation is rewritten in an inclusion-exclusion fashion as

$$d - \|x_1\| - \|x_2\| - \|x_3\| + \|x_1 \wedge x_2\| + \|x_1 \wedge x_3\| + \|x_2 \wedge x_3\| - \|x_1 \wedge x_2 \wedge x_3\| = 0$$

(here, $d = \| \bigwedge_{i \in \emptyset} x_i \|$).

However, even though we can efficiently determine all values $\| \bigwedge_{i \in I} x_i \|$ in time $O(m^{|I|})$, computing $\| \bigwedge_{i=1}^{k} x_i \|$ is infeasible if we want to beat the baseline algorithm. Therefore, our next algorithm makes use of another trick: When combining two equations – as in the NAE example – we can sometimes exploit cancellations to decide instances without actually computing $\| \bigwedge_{i=1}^{k} x_i \|$: Because the Hamming weight of all vectors is always non-negative, instead of testing $\|x_1 \wedge x_2 \wedge x_3\| = 0$ and $\|\bar{x}_1 \wedge \bar{x}_2 \wedge \bar{x}_3\| = 0$, we can equivalently test whether $\|x_1 \wedge x_2 \wedge x_3\| + \|\bar{x}_1 \wedge \bar{x}_2 \wedge \bar{x}_3\| = 0$. By expanding $\|\bar{x}_1 \wedge \bar{x}_2 \wedge \bar{x}_3\|$ as above, the $\|x_1 \wedge x_2 \wedge x_3\|$ term cancels and it suffices to know the numbers of all I-stars, for $I \subsetneq [3]$.

▶ **Lemma 13.** *Let $\alpha, \beta \in \{0,1\}^3$ be of odd Hamming distance. In time $O(m^2)$, we can determine the edge weights of a* Sum *constraint C such that $(\forall y)\, (x_1[y], x_2[y], x_3[y]) \notin \{\alpha, \beta\}$ holds if and only if (x_1, x_2, x_3) satisfies C.*

Proof. Let $\alpha = (\alpha_1, \alpha_2, \alpha_3)$ and $\beta = (\beta_1, \beta_2, \beta_3)$. Without loss of generality, we may assume that, for some $i^* \in \{1, 3\}$, $\alpha_i \neq \beta_i$ for all $i \leq i^*$ and $\alpha_i = \beta_i$ for all $i > i^*$.

As argued before, any triple (x_1, x_2, x_3) satisfies $(\forall y)\, (x_1[y], x_2[y], x_3[y]) \notin \{\alpha, \beta\}$ if and only if (x_1, x_2, x_3) satisfies $\|x_1^{\alpha_1} \wedge x_2^{\alpha_2} \wedge x_3^{\alpha_3}\| = \|x_1^{\beta_1} \wedge x_2^{\beta_2} \wedge x_3^{\beta_3}\| = 0$. Since both sides of the left equation are always non-negative, we can equivalently demand that their sum be zero. This condition simplifies to:

$$\begin{aligned}
0 &= \|x_1^{\alpha_1} \wedge x_2^{\alpha_2} \wedge x_3^{\alpha_3}\| + \|x_1^{\beta_1} \wedge x_2^{\beta_2} \wedge x_3^{\beta_3}\| \\
&= \|(\bigwedge_{i>i^*} x_i^{\alpha_i}) \wedge (\bigwedge_{i \leq i^*} x_i^{\alpha_i})\| + \|(\bigwedge_{i>i^*} x_i^{\alpha_i}) \wedge (\bigwedge_{i \leq i^*} \bar{x}_i^{\alpha_i})\| \\
&= \|(\bigwedge_{i>i^*} x_i^{\alpha_i}) \wedge (\bigwedge_{i \leq i^*} x_i^{\alpha_i})\| + \sum_{I \subseteq [i^*]} (-1)^{|I|} \|(\bigwedge_{i>i^*} x_i^{\alpha_i}) \wedge (\bigwedge_{i \in I} x_i^{\alpha_i})\| && \text{Observation 2} \\
&= \sum_{I \subsetneq [i^*]} (-1)^{|I|} \|(\bigwedge_{i>i^*} x_i^{\alpha_i}) \wedge (\bigwedge_{i \in I} x_i^{\alpha_i})\|. && i^* \text{ is odd}
\end{aligned}$$

Notice that (possibly after applying Observation 2 again to get rid of complemented occurrences) this expression is a weighted sum of values d, $\|x_1\|$, $\|x_2\|$, $\|x_3\|$, $\|x_1 \wedge x_2\|$, $\|x_1 \wedge x_3\|$ and $\|x_2 \wedge x_3\|$. In particular, the problematic term $\|x_1 \wedge x_2 \wedge x_3\|$ canceled in the equation above due to i^* being an odd number. In summary, we can express the above constraint by $0 = \sum_{I \subsetneq [3]} \lambda_I \| \bigwedge_{i \in I} x_i \|$ for some constants λ_I.

It is easy to compute the values $\| \bigwedge_{i \in I} x_i \|$ for all $I \subsetneq [3]$: $\| \bigwedge_{i \in \emptyset} x_i \| = d$ is a constant and each number $\|x_i\|$ is obtained by once enumerating all edges between X_i and Y. Any value $\|x_i \wedge x_j\|$ is determined in time $O(m^2)$ by iterating over all pairs of edges between X_i and Y, and between X_j and Y, respectively. Finally, we annotate each edge of the Sum constraint accordingly. To this end, each edge (x_i, x_j) is labeled with the contribution of

$\|x_i \wedge x_j\|$ and additionally, we distribute the node-weights $\|x_1\|$, $\|x_2\|$ and $\|x_3\|$ to the edges:

$$w(x_1, x_2) := \lambda_{\{1,2\}}\|x_1 \wedge x_2\| + \lambda_{\{1\}}\|x_1\|,$$
$$w(x_2, x_3) := \lambda_{\{2,3\}}\|x_2 \wedge x_3\| + \lambda_{\{2\}}\|x_2\|,$$
$$w(x_3, x_1) := \lambda_{\{3,1\}}\|x_3 \wedge x_1\| + \lambda_{\{3\}}\|x_3\|.$$

The target t is set to $-\lambda_\emptyset d$.

It is left to show that $\sum_{e\in E} |w(e)| \leq O(m^2)$ as in the definition of Sum. It is clear that $\|x_1\|$, $\|x_2\|$ and $\|x_3\|$ only account for $O(m)$. Each edge (x_i, y) contributes to at most m values $\|x_i \wedge x_j\|$, thus $\sum_{x_i,x_j} \|x_i \wedge x_j\| \leq O(m^2)$. The values λ_I are fixed constants depending only on α, β. ◀

4.2.3 Combined Algorithm for $k = 3$

By combining the previous reductions from VP(ϕ) to Constrained Triangle problems, we find the desired algorithm for 3-variable Vector Problems of hardness at most 2:

▶ **Lemma 14.** *Let Φ be a set of 3-variable formulas of hardness $H(\Phi) \leq 2$. Then* VP(Φ) *is decidable in time $O(m^{3-\varepsilon})$ for some $\varepsilon > 0$ using fast matrix multiplication.*

Proof. Let $\phi \in \Phi$. Given an VP(ϕ) instance, we can obtain an equivalent Constrained Triangle formulation by covering each pair of falsifying assignments by an Equal or Sum constraint using Lemmas 12 and 13 (any pair of distinct 3-variable assignments is either of Hamming distance 2 or of odd Hamming distance).

The hybrid problem VP(Φ) is now solved by rephrasing each subproblem VP(ϕ), $\phi \in \Phi$, as a Constrained Triangle problem Triangle$[C_1^\phi] \cdots [C_{r_\phi}^\phi]$ with $C_i^\phi \in \{\text{Equal}, \text{Sum}\}$, and by "stacking" all constraints. Since $\Phi = \{\phi_1, \ldots, \phi_\ell\}$ is of constant size, we are left to solve an instance of Triangle$[C_1^{\phi_1}] \cdots [C_{r_{\phi_1}}^{\phi_1}] \cdots [C_1^{\phi_\ell}] \cdots [C_{r_{\phi_\ell}}^{\phi_\ell}]$. Lemma 8 yields an $O(m^{3-\varepsilon})$-time algorithm for some $\varepsilon > 0$ for any constant number of adjunct Equal and Sum constraints. ◀

4.3 Algorithms for Arbitrary k

For large k, we will tackle Vector Problems by first brute-forcing over a number of variables before solving the remaining $k' = 2$ or $k' = 3$ problem. To this end, we establish the following lemmas in order to lift fast algorithms from fewer quantifiers to more quantifiers:

▶ **Lemma 15.** *Let ϕ be a k-variable formula. Then, for any $k' \leq k$, there exists some $I' \in \binom{[k]}{k-k'}$ such that for any I'-restriction ϕ' of ϕ we have $H(\phi') \leq H(\phi)$.*

Proof. Let $h = H(\phi)$. The statement is clear for $k' \leq h$, so suppose $h < k' \leq k$. From $H(\phi) < h + 1$, it follows by definition that there exists some $I \in \binom{[k]}{k-h-1}$ such that any I-restriction ϕ' of ϕ has not exactly one falsifying assignment. Choose an arbitrary $I' \in \binom{I}{k-k'}$ and let ϕ' be any I'-restriction of ϕ. Then any $(I \setminus I')$-restriction of ϕ' has not exactly one falsifying assignment. Hence, $H(\phi') < k' - |I \setminus I'| = k' - (k - h - 1) + (k - k') = h + 1$ and thus $H(\phi') \leq h = H(\phi)$. ◀

▶ **Lemma 16** (Lifting). *Let $k' \leq k$, let ϕ be a k-variable formula and let Φ' contain all k'-variable formulas ϕ' of hardness $H(\phi') \leq H(\phi)$. If* VP(Φ') *is decidable in time $T(m)$, then* VP(ϕ) *is decidable in time $O(m^{k-k'}T(m))$.*

Proof. The idea is to appropriately brute-force over $k - k'$ variables $x_1, \ldots, x_{k-k'}$ and use the $\mathrm{VP}(\Phi')$ algorithm to decide the remaining instance. Here, it is crucial that we can indeed solve the *Hybrid* Vector Problem! This is because we have to specialize ϕ to the values of the brute-forced vectors $(x_1, \ldots, x_{k-k'})$, however, in general $(x_1[y], \ldots, x_{k-k'}[y])$ takes different values for different y's, so we have to instantiate ϕ "per dimension". This leads to queries allowing several formulas ϕ_y – exactly as permitted by Hybrid Vector Problems.

More formally, by Lemma 15, there exists some index set I' of size $k - k'$, such that any I'-restriction ϕ' of ϕ satisfies $H(\phi') \leq H(\phi)$. By interchanging the order of the existential variables, we can assume that $I' = \{1, \ldots, k - k'\}$. Our first step is to enumerate all $n^{k'-k}$ choices for the first $k - k'$ variables $(x_1, \ldots, x_{k-k'})$; fix such a tuple $(x_1, \ldots, x_{k-k'})$. It remains to solve the problem

$$(\exists x_{k-k'+1}) \ldots (\exists x_k)\, (\forall y)\, \phi_y(x_{k-k'+1}[y], \ldots, x_k[y]),$$

where ϕ_y denotes the I'-restriction of $\phi(x_1[y], \ldots, x_k[y])$ in which all occurrences of $x_1[y], \ldots, x_{k-k'}[y]$ are fixed as specified by the brute-forced vectors $x_1, \ldots, x_{k-k'}$. In other words, we are left to solve a Hybrid Vector Problem over the formula set $\bigcup_y \{\phi_y\}$. Recall that ϕ_y is an I'-restriction of ϕ, and therefore, as guaranteed by Lemma 15, $H(\phi_y) \leq H(\phi)$. Thus $\phi_y \in \Phi'$, by the definition of Φ'. By the assumption that we can solve $\mathrm{VP}(\Phi')$ in time $T(m)$, the total running time is $O(n^{k-k'} T(m))$, which is bounded by $O(m^{k-k'} T(m))$. ◀

By the Lifting Lemma, we conclude the following consequences of Lemma 11 and Lemma 14. This concludes the algorithmic part of Theorem 6.

▶ **Corollary 17.** *Let ϕ be a k-variable formula of hardness $H(\phi) \leq 1$. Then $\mathrm{VP}(\phi)$ is decidable in time $O(m^{k-1})$.*

▶ **Corollary 18.** *Let ϕ be a k-variable formula of hardness $H(\phi) \leq 2 < k$. Then $\mathrm{VP}(\phi)$ is decidable in time $O(m^{k-\varepsilon})$ for some $\varepsilon > 0$ using fast matrix multiplication.*

5 Hardness Results

In this section, we prove the hardness part of Theorem 6, see Lemmas 19 and 20, below.

Let ϕ be a k-variable formula with exactly one falsifying assignment, or equivalently, let ϕ be of hardness $H(\phi) = k$. The model-checking query $(\exists x_1) \ldots (\exists x_k)\, (\forall y)\, \phi(x_1[y], \ldots, x_k[y])$ – that is, $\mathrm{VP}(\phi)$ – is equivalent to the k-Orthogonal Vectors problem with respect to polynomial improvements over the $O(m^k)$-time baseline algorithm as shown by Gao et al. [41] (the authors of [41] refer to Vector Problems of hardness k as *Basic Problems*). This shows the k-OV hardness part of Theorem 6:

▶ **Lemma 19** ([41, Lemma 1.1, Lemma 5.1])**.** *Let $H(\phi) = k$. If $\mathrm{VP}(\phi)$ is decidable in time $O(m^{k-\varepsilon})$ for some $\varepsilon > 0$, then k-OV is decidable in time $O(n^{k-\varepsilon'} \operatorname{poly} d)$ for some $\varepsilon' > 0$.*

It remains to show the HyperClique hardness. Recall that given a k-partite h-uniform hypergraph $(V_1 \uplus \ldots \uplus V_k, E)$, the h-uniform k-HyperClique problem asks to find vertices $(v_1, \ldots, v_k) \in V_1 \times \ldots \times V_k$, so that any h vertices in $\{v_1, \ldots, v_k\}$ form a hyperedge in E.

▶ **Lemma 20.** *Let ϕ be a k-variable formula and let $2 \leq h \leq H(\phi)$. If $\mathrm{VP}(\phi)$ is decidable in time $O(m^{k-\varepsilon})$ for some $\varepsilon > 0$, then the h-uniform HyperClique hypothesis fails.*

For $h \geq 3$, Lemma 20 shows the hardness part of Theorem 6. Moreover, the special case $h = 2$ is also informative: The 2-uniform k-HyperClique problem is simply the k-Clique problems on ordinary graphs. It is well-known that k-Clique can be solved in time $O(n^{\frac{\omega k}{3}})$

using fast matrix multiplication. However, no combinatorial solution which is significantly faster than the $O(n^k)$-time exhaustive search approach is known. The combinatorial k-Clique hypothesis speculates that no polynomial improvement is possible without relying on algebraic methods [3], which in turn, confirms the need to fall back on fast matrix multiplication in Lemma 14.

Proof. We will fix $k' = k'(k,h,\varepsilon)$ later such that k divides k'. Given an h-uniform k'-HyperClique instance G', we can assume that G' is k'-partite by copying the vertex set k' times and by keeping only edges spanned between distinct copies. So let $V(G') = V'_{1,1} \uplus \ldots \uplus V'_{1,k'/k} \uplus \ldots \uplus V'_{k,1} \uplus \ldots \uplus V'_{k,k'/k}$. We proceed in a split-and-list fashion: First, we split $V(G')$ into the k parts $V'_i := V'_{i,1} \uplus \ldots \uplus V'_{i,k'/k}$. Then, for $i = 1, \ldots, k$, we let $X_i \subseteq V'_{i,1} \times \ldots \times V'_{i,k'/k}$ be all tuples of vertices that form an h-uniform clique in G'. We refer to the elements of X_i as *bundles*. Let $Y \subseteq \binom{V(G')}{h}$ contain all non-edges of G'. We say a vertex bundle $x_i \in X_i$ *avoids* a non-edge $y \in Y$ if not all vertices in $y \cap V'_i$ are contained in x_i.

Our next step is to assign the entries $x_i[y]$, for all $x_i \in X_i$ and $y \in Y$. Let $y = \{v_1, \ldots, v_h\} \in Y$ and collect all indices $J = \{j : v_i \in V'_j \text{ for some } i \in [h]\}$. Since ϕ is of hardness $H(\phi) \geq h$, it holds for all index sets I of size at least $k - h$ that there exists an I-restriction ϕ' of ϕ having exactly one falsifying assignment. Picking $I = [k] \setminus J$, there exists an I-restriction $\phi' = \phi|_\alpha$, for some $\alpha : I \to \{0,1\}$, such that ϕ' is falsified only by a single assignment $\beta : J \to \{0,1\}$. For all $i \in I$ and all vectors $x_i \in X_i$, we define $x_i[y] := \alpha_i$. For all $j \in J$ and all vectors $x_j \in X_j$, we define $x_j[y] := \bar{\beta}_j$ if x_j avoids y, and $x_j[y] := \beta_j$ otherwise.

We claim that there exists a tuple of vectors $(x_1, \ldots, x_k)$ with $(\forall y \in Y)\, \phi(x_1[y], \ldots, x_k[y])$ if and only if G' contains a k'-hyperclique. Suppose there exists some k'-hyperclique $(v_{1,1}, \ldots, v_{1,k'/k}, \ldots, v_{k,1}, \ldots, v_{k,k'/k})$ in G'. Since each tuple $(v_{i,1}, \ldots, v_{i,k'/k})$ by itself forms a hyperclique, we have that $x_i := (v_{i,1}, \ldots, v_{i,k'/k}) \in X_i$. We aim to prove that $(x_1, \ldots, x_k)$ satisfies $\phi(x_1[y], \ldots, x_k[y])$ for all $y \in Y$. Let $y \in Y$ be arbitrary, and let J, I, α and β as above. After plugging in all values $x_i[y] = \alpha_i$ for $i \in I$, there remains only one falsifying assignment of $\phi(x_1[y], \ldots, x_k[y])$, given by β. Since we started from a hyperclique, x_j must avoid y for some $j \in J$, and thus $x_j = \bar{\beta}_j$ for some $j \in J$. The vectors $(x_1, \ldots, x_k)$ thus satisfy $\phi(x_1[y], \ldots, x_k[y])$, for any y. The converse argument is essentially symmetric. This finishes the correctness proof.

Finally, we turn to the running time analysis. Observe that $|X_i| \leq O(n^{k'/k})$ for all i and $|Y| \leq O(n^h)$. Furthermore, constructing the X_i's and Y takes time $O(n^{k'/k})$. Assigning the entries $x_i[y]$ takes time $O(\sum_i |X_i| \cdot |Y|) = O(n^{k'/k+h})$. In particular, it follows that the number of edges is bounded by $m \leq O(n^{k'/k+h})$. By assumption, solving $\mathrm{VP}(\phi)$ is in time $O(m^{k-\varepsilon})$ for some $\varepsilon > 0$. Hence, in total we solve h-uniform k'-HyperClique in time $O(n^{(k'/k+h)(k-\varepsilon)})$. By setting $k' \geq hk^2/\varepsilon$ this is bounded by $O(n^{k'-h\varepsilon})$, contradicting the h-uniform HyperClique hypothesis. ◀

We remark that for $h = k$, Lemma 20 can also be derived in an alternative way: Abboud et al. [5] show that the k-OV hypothesis is implied by the h-uniform HyperClique hypothesis for all $h \geq 3$. So all we need to do is combine Lemma 19 with that reduction:

▶ **Lemma 21** ([5]). *If k-OV is decidable in time $O(n^{k-\varepsilon} \operatorname{poly} d)$ for some $\varepsilon > 0$, then, for any h, the h-uniform HyperClique hypothesis fails.*

This finishes the proof of Theorem 6.

6 Equivalence of Vector Problems and $\exists^k\forall$ Graph Properties

This section is devoted to proving that Vector Problems $\mathrm{VP}(\phi)$ capture the core difficulty of model-checking $\exists^k\forall$ graph properties. Specifically, we will prove Theorem 7, which together with Theorem 6 proves Theorems 1 and 3.

▶ **Theorem 7** (Restated). *Let $\psi = (\exists x_1)\dots(\exists x_k)\,(\forall y)\,\phi(x_1,\dots,x_k,y)$ be an $\exists^k\forall$ graph property and let ϕ_0 denote the formula ϕ after substituting each predicate $E(x_i,x_j)$ by false.*

- *If $\mathrm{MC}(\psi)$ is decidable in time $T(m)$, then $\mathrm{VP}(\phi_0)$ is decidable in time $O(T(m))$.*
- *If $\mathrm{VP}(\phi_0)$ is decidable in time $O(m^{k-\varepsilon})$ for some $\varepsilon > 0$, then $\mathrm{MC}(\psi)$ is decidable in time $O(m^{k-\varepsilon'})$ for some $\varepsilon' > 0$.*

Recall that $\exists^k\forall$ graph properties are strictly more general than Vector Problems in the sense that graph properties are sensitive to edges $E(x_i,x_j)$. We will continue to interchangeably interpret the x_i's as vertices and vectors depending on the context. For the remainder of this section, let ψ, ϕ and ϕ_0 be as in Theorem 7.

We start by demonstrating the following preliminary lemma:

▶ **Lemma 22.** *Given a set of two-element subsets $\emptyset \neq I \subseteq \binom{[k]}{2}$, we say that vertices $(x_1,\dots,x_k)$* respect *I if for any $i<j$ we have $E(x_i,x_j)$ if and only if $\{i,j\} \in I$. We can compute the values $\|\bigwedge_{i=1}^{k} x_i\|$ for all vectors $(x_1,\dots,x_k)$ respecting I in time $O(m^{k-\frac{1}{2}})$.*

Proof. Since I is non-empty, there exists at least some element $\{i,j\} \in I$. We start by brute-forcing over all $k-2$ vectors x_ℓ, $\ell \notin \{i,j\}$. As in the statement, we only care about the combinations that respect I. In the same manner, we only keep the vertices x_i, x_j consistent with I. In particular, for all x_i, x_j overcoming this step, (x_i,x_j) indeed forms an edge. It is enough to compute $\|x_i \wedge x_j\|$ for these (x_i,x_j) in time $O(m^{\frac{3}{2}})$, because thereby we achieve a total running time of $O(n^{k-2}m^{\frac{3}{2}}) = O(m^{k-\frac{1}{2}})$. In other words, we are left to count, for each pair (x_i,x_j), the number of triangles containing (x_i,x_j) in the remaining graph with partitions X_i, X_j and Y. For the sake of completeness, we proceed to sketch the well-known procedure of counting triangles in sparse tripartite graphs [12].

We call a vertex *heavy* if it is of degree at least m^δ (where δ is yet to be fixed), and *light* otherwise. All light vertices can be eliminated as follows: Enumerate all edges $\{u,v\}$ and if v is light, then further iterate over all edges $\{v,w\}$ such that u, v and w stem from different partitions. Remember each triangle found in that manner and remove all light vertices afterwards. This step accounts for $O(m \cdot m^\delta) = O(m^{1+\delta})$ time.

The remaining graph consists only of heavy vertices, and, since there are only m edges, at most $O(m/m^\delta) = O(m^{1-\delta})$ such. We may apply the naïve algorithm by explicitly considering each triple of vertices. This step takes time $O(m^{3(1-\delta)})$, so in total the algorithm runs in time $O(m^{1+\delta} + m^{3(1-\delta)})$. The claim follows by setting $\delta := \frac{1}{2}$.

We remark that using fast matrix multiplication to solve the instance including only heavy vertices, we can achieve a slightly faster algorithm running in total time $O(m^{k-2+\frac{2\omega}{\omega+1}})$. ◀

Proof of Theorem 7. The first part is easy to see: The Vector Problem $\mathrm{VP}(\phi_0)$ constitutes a special case of model-checking ψ.

So let us focus on the second part. For a set $I \subseteq \binom{[k]}{2}$, let ϕ_I denote ϕ after substituting each predicate $E(x_i,x_j)$ by *true* if $\{i,j\} \in I$ and by *false* otherwise. In particular, $\phi_\emptyset = \phi_0$. Furthermore, we define

$$\psi_I := (\exists x_1)\dots(\exists x_k)\left(\left(\bigwedge_{\{i,j\}\in\binom{[k]}{2}} E(x_i,x_j) \leftrightarrow \big[\{i,j\} \in I\big]\right) \wedge (\forall y)\,\phi_I(x_1[y],\dots,x_k[y])\right).$$

Then clearly $\psi = \bigvee_{I \subseteq \binom{[k]}{2}} \psi_I$. So we can check the satisfiability of ψ by separately model-checking all properties ψ_I.

We claim that model-checking ψ_I for any $I \neq \emptyset$ is in time $O(m^{k-\frac{1}{2}})$: Observation 3 shows how to test, for a fixed tuple $(x_1, \dots, x_k)$ and given the values $\| \bigwedge_{j \in J} x_j \|$ for all $J \subseteq [k]$, whether it holds that $(\forall y)\, \phi_I(x_1[y], \dots, x_k[y])$. We thus start by precomputing all values $\| \bigwedge_{j \in J} x_j \|$ for all sets $J \subsetneq [k]$. To this end, enumerate all $|J|$-tuples of edges between X_j and Y, $j \in J$; this step takes time $O(m^{k-1})$. Using Lemma 22 we can further precompute $\| \bigwedge_{j \in [k]} x_j \|$ for all vertices $(x_1, \dots, x_k)$ respecting I in time $O(m^{k-\frac{1}{2}})$. Together, we now know $\| \bigwedge_{j \in J} x_j \|$ for all J and all vertices $(x_1, \dots, x_k)$ respecting I. Hence, it suffices to enumerate all vertices $(x_1, \dots, x_k)$ respecting I and to test as in Observation 3 whether $(\forall y)\, \phi_I(x_1[y], \dots, x_k[y])$ holds (this check takes constant time). Since there are at most $O(m^{k-1})$ many tuples $(x_1, \dots, x_k)$ respecting I, the running time is dominated by $O(m^{k-\frac{1}{2}})$.

It remains to find a model-checking procedure for $\psi_\emptyset$, i.e., for $I = \emptyset$. Note that $\mathrm{MC}(\psi_\emptyset)$ is not directly equivalent to $\mathrm{VP}(\phi_\emptyset)$, since an arbitrary solution $(x_1, \dots, x_k)$ of $\phi_\emptyset$ does not necessarily meet the condition that none of the edges (x_i, x_j) are present. The following reduction enforces this constraint.

Consider a given $\mathrm{MC}(\psi_\emptyset)$ instance G over the vertex partitions $X_1, \dots, X_k$ and Y and let $\delta > 0$ be a parameter to be fixed later. We call a vertex x_i *heavy* if it is of degree $\geq m^\delta$, and *light* otherwise. The first step is to eliminate all heavy vertices; there can exist at most $O(m/m^\delta) = O(m^{1-\delta})$ many such vertices. By interchanging the order of the existential quantifiers, we can always assume that x_1 is heavy and solve the remaining problem over $X_2, \dots, X_k$ in time $O(m^{k-1})$ using the model-checking baseline algorithm. If a solution $(x_1, \dots, x_k)$ is found in that manner, we accept. It thus takes time $O(m^{k-\delta})$ to safely remove all heavy vertices.

Next, partition each set X_i into several groups $X_{i,1}, \dots, X_{i,g}$ such that the total degree of all vectors is bounded by $m^\delta \leq \sum_{x_i \in X_{i,j}} \deg(x_i) \leq 2m^\delta$, for all groups $X_{i,j}$, except for possibly the last non-empty groups. This is implemented by greedily inserting vectors into $X_{i,j}$ until its total degree exceeds m^δ. As each vector inserted in that way is light, we can overshoot by at most m^δ. It follows that $g \leq O(m/m^\delta) = O(m^{1-\delta})$.

We assume that $\mathrm{VP}(\phi_0) = \mathrm{VP}(\phi_\emptyset)$ is decidable in time $O(m^{k-\varepsilon})$ for some $\varepsilon > 0$. Then we continue as follows:

1. For all combinations $(j_1, \dots, j_k) \in [g]^k$, solve the Vector Problem $\mathrm{VP}(\phi_\emptyset)$ with input $X_{1,j_1}, \dots, X_{k,j_k}$. If we find a solution, we call $(j_1, \dots, j_k)$ a *successful* combination.
2. If there are more than m^{k-1} successful combinations, we accept.
3. Otherwise, for any successful combination $(j_1, \dots, j_k)$, solve $\mathrm{MC}(\psi_\emptyset)$ using the baseline algorithm on $X_{1,j_1}, \dots, X_{k,j_k}$ and accept iff one of these invocations accepted.

We claim the above algorithm is correct. First of all, any solution $(x_1, \dots, x_k)$ of $\mathrm{MC}(\psi_\emptyset)$ is also a solution of $\mathrm{VP}(\phi_\emptyset)$. It is therefore safe to only consider those subinstances in step 3 for which we received a positive output in step 1. It remains to argue why step 2 is correct. How many tuples $(x_1, \dots, x_k)$ can be solutions of $\mathrm{VP}(\phi_\emptyset)$ but not of $\mathrm{MC}(\psi_\emptyset)$? At most $mn^{k-2} \leq m^{k-1}$, since at least one edge (x_i, x_j) must exist for any such tuple. Thus, if we witness $> m^{k-1}$ solutions of $\mathrm{VP}(\phi_\emptyset)$, among these there exists at least one solution of $\mathrm{MC}(\psi_\emptyset)$ by guarantee.

Finally, let us bound the running time of the above algorithm. Recall that removing heavy vertices accounts for $O(m^{k-\delta})$ time. In step 1, the $\mathrm{VP}(\phi_\emptyset)$ algorithm is applied $g^k = O(m^{k-\delta k})$ times on instances of size $O(m^\delta)$, which takes time $O(m^{k-\delta k+\delta(k-\varepsilon)}) = O(m^{k-\delta\varepsilon})$. Step 3 becomes relevant only if there are at most m^{k-1} successful combinations. For any

such combination, the model-checking baseline algorithm takes time $O((m^\delta)^k) = O(m^{\delta k})$. In total, our running time is $O(m^{k-\delta} + m^{k-\delta\varepsilon} + m^{k-1+\delta k})$. By picking δ so that $0 < \delta < \frac{1}{k}$, the claim follows.

This finishes the proof of Theorem 7, and thus completes the proof of our main result. ◀

7 Extensions and Outlook

Beyond our results of Theorems 1 and 3, we discuss several natural directions for extensions, present first results along these lines and give open problems for future work. In particular, we extend our results to a counting dichotomy and investigate the optimal exponent of low-complexity properties.

7.1 Determining the Optimal Exponent for Low-Complexity Properties

For the optimal exponent c_ψ for any $\exists^k\forall$-quantified graph property ψ, Theorems 1 and 3 establish either a (conditionally) tight value of $c_\psi = k$ or an upper bound of $c_\psi < k$. This begs the question: Can we obtain the (conditionally) exact value on c_ψ also in the latter case?

As an interesting exemplary case, we study Vector Problems $(\exists x_1 \in X_1)\,(\exists x_2 \in X_2)\,(\exists x_3 \in X_3)\,(\forall y \in Y)\,\phi(x_1[y], x_2[y], x_3[y])$ where ϕ is symmetric, i.e., $\phi(x_1, x_2, x_3)$ is invariant under interchanging the variables' order. Equivalently, ϕ is symmetric if its output depends only on the number of 1-inputs. We therefore identify a 3-variable symmetric formula with its *symmetric type*, a zero-based length-4 string $t \in \{\mathbf{0}, \mathbf{1}\}^4$ where $t_i = \mathbf{1}$ exactly if ϕ holds true on all inputs of i 1's and $(3 - i)$ 0's.

For symmetric formulas ϕ, we find a more immediate criterion to read off the hardness $H(\phi)$. Namely, $H(\phi)$ equals the maximum number h, such that $\mathbf{1}^h\mathbf{0}$ or $\mathbf{01}^h$ is a substring[6] of ϕ's symmetric type (and $H(\phi) = 0$ if neither constitutes a substring).

▶ **Theorem 7.** *Let $\phi(x_1, x_2, x_3)$ be symmetric. The complexity of* $\mathrm{VP}(\phi)$ *is as stated in Table 1.*

We prove Theorem 7 in the full version of this paper. As detailed there (and illustrated by Table 1), already in this exemplary case some interesting gaps remain and offer potential starting points for future work.

7.2 Counting Classification

For first-order graph properties with our quantifier structure $\exists^k\forall$, it is natural to ask if we can *count* the number of its witnesses. Specifically, for a given property $\psi = (\exists x_1 \in X_1)\ \ldots\ (\exists x_k \in X_k)\,(\forall y \in Y)\,\phi(x_1, \ldots, x_k, y)$, we might ask to output the number of tuples $(x_1, \ldots, x_k)$ such that $(\forall y)\,\phi(x_1, \ldots, x_k, y)$ holds (instead of merely detecting existence of at least one such tuple) – we call this problem the counting model-checking problem $\#\mathrm{MC}(\psi)$. Especially in the context of database queries, one often would like to report this number or even enumerate all such tuples. Indeed, related work [36] considers such questions for more general quantifier structures, but under other restrictions on the formulas and when the running time is measured in terms of the number of objects n rather than m.

Note that the analogous question for Boolean constraint satisfaction properties is resolved: For every Boolean CSP, we can either count the number of solutions in polynomial-time, or this task is #P-complete [33]. In our case, this dichotomy is a surprisingly simple consequence of our techniques. In particular, we achieve the same dichotomy as for the decision version.

[6] A contiguous sequence of characters within ϕ's symmetric type

Table 1 Lists all symmetric 3-variable formulas ϕ and the complexities of the respective Vector Problems VP(ϕ).

hardness $H(\phi)$	*symmetric type of* ϕ	*upper bound*	*lower bound*
0	**0000**	trivial	trivial
0	**1111**	trivial	trivial
1	**0001**, **1000**	$O(m)$	$\Omega(m)$
1	**0010**, **0100**	$O(m^2)$	$\Omega(m)$
1	**0101**, **1010**	$O(m^2)$	$m^{2-o(1)}$ under 3-XOR
1	**1001**	$O(m)$	$\Omega(m)$
2	**0011**, **1100**	$O(m^\omega)$	$m^{\omega-o(1)}$ under k-Clique
2	**0110**	$\tilde{O}(m^{\frac{3+\omega}{2}})$	$m^{\omega-o(1)}$ under k-Clique
2	**1011**, **1101**	$O(m^{\frac{9+\omega}{4}})$	$m^{\omega-o(1)}$ under k-Clique
3	**0111**, **1110**	$m^3/2^{\Omega(\sqrt{\log m})}$	$m^{3-o(1)}$ under 3-OV

▶ **Theorem 7.** *Let ψ be an $\exists^k\forall$ graph property of hardness $h = H(\psi)$.*

- *If $h \leq 1$, then $\#\mathrm{MC}(\psi)$ is solvable in time $O(m^{k-\varepsilon})$ for some $\varepsilon > 0$ combinatorially.*
- *If $h \leq 2 < k$, then $\#\mathrm{MC}(\psi)$ is solvable in time $O(m^{k-\varepsilon})$ for some $\varepsilon > 0$ using fast matrix multiplication. (Furthermore, $\#\mathrm{MC}(\psi)$ cannot be solved by a combinatorial $O(m^{k-\varepsilon})$-time algorithm unless the combinatorial k-Clique hypothesis is false.)*
- *If $3 \leq h \leq k$, then $\#\mathrm{MC}(\psi)$ cannot be solved in time $O(m^{k-\varepsilon})$ for any $\varepsilon > 0$ unless the h-uniform HyperClique hypothesis fails.*
- *If $h = k$, then $\#\mathrm{MC}(\psi)$ cannot be solved in time $O(m^{k-\varepsilon})$ for any $\varepsilon > 0$ unless the k-OV hypothesis fails.*

We defer the proof of this theorem to the full version of this paper.

7.3 Open Problems

Among the natural remaining challenges is, first and foremost, the following: Can we generalize or strengthen our dichotomy to larger fragments of first-order properties?

Specifically, one direction is to extend our results beyond graph properties: Allowing more than a single predicate or allowing a single predicate of higher arity already yields further very expressive classes. While our algorithmic techniques conveniently generalize, unfortunately, they do not yet seem to be sufficient to establish a complete dichotomy.

A second direction is to investigate other quantifier structures than $\exists^k\forall$ (and the equivalent $\forall^k\exists$). Note that establishing such a dichotomy might require different plausible hardness assumptions than the ones used in this work – in particular, it follows from the work of Carmosino et al. [28] that any such hardness assumption must have a lower nondeterministic *and* co-nondeterministic complexity than its deterministic complexity.

Finally, it is interesting to explore whether Proposition 4 can be strengthened: In particular, assume that for some hardness level $h \geq 3$, there is an algorithm that allows us to detect a k-clique in h-uniform hypergraphs in time $O(n^{k-\varepsilon})$ for all $k \geq h+1$. Can we then solve *all* hardness-h $\exists^k\forall$ graph properties in time $O(m^{k-\varepsilon'})$?

References

1. Amir Abboud, Arturs Backurs, Karl Bringmann, and Marvin Künnemann. Fine-Grained Complexity of Analyzing Compressed Data: Quantifying Improvements over Decompress-and-Solve. In *Proceedings of the 2017 IEEE 58th Annual Symposium on Foundations of Computer Science*, FOCS '17, pages 192–203, 2017. doi:10.1109/FOCS.2017.26.
2. Amir Abboud, Arturs Backurs, Thomas Dueholm Hansen, Virginia Vassilevska Williams, and Or Zamir. Subtree Isomorphism Revisited. *ACM Trans. Algorithms*, 14(3):27:1–27:23, 2018. doi:10.1145/3093239.
3. Amir Abboud, Arturs Backurs, and Virginia Vassilevska Williams. If the Current Clique Algorithms Are Optimal, So is Valiant's Parser. In *Proceedings of the 2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, FOCS '15, pages 98–117, 2015. doi:10.1109/FOCS.2015.16.
4. Amir Abboud, Arturs Backurs, and Virginia Vassilevska Williams. Tight Hardness Results for LCS and Other Sequence Similarity Measures. In *Proceedings of the 2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, FOCS '15, pages 59–78, 2015. doi:10.1109/FOCS.2015.14.
5. Amir Abboud, Karl Bringmann, Holger Dell, and Jesper Nederlof. More Consequences of Falsifying SETH and the Orthogonal Vectors Conjecture. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, pages 253–266. ACM, 2018. doi:10.1145/3188745.3188938.
6. Amir Abboud, Virginia Vassilevska Williams, and Joshua R. Wang. Approximation and Fixed Parameter Subquadratic Algorithms for Radius and Diameter in Sparse Graphs. In *Proceedings of the 26th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '16, pages 377–391, 2016. doi:10.1137/1.9781611974331.ch28.
7. Amir Abboud, Ryan Williams, and Huacheng Yu. More Applications of the Polynomial Method to Algorithm Design. In *Proceedings of the 26th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '15, pages 218–230, 2015. URL: http://dl.acm.org/citation.cfm?id=2722129.2722146.
8. Amir Abboud, Virginia Vassilevska Williams, and Oren Weimann. Consequences of Faster Alignment of Sequences. In Javier Esparza, Pierre Fraigniaud, Thore Husfeldt, and Elias Koutsoupias, editors, *Automata, Languages, and Programming*, pages 39–51. Springer Berlin Heidelberg, 2014.
9. Serge Abiteboul, Richard Hull, and Victor Vianu, editors. *Foundations of Databases: The Logical Level*. Addison-Wesley Longman Publishing Co., Inc., 1st edition, 1995.
10. Alfred V. Aho, Yehoshua Sagiv, and Jeffrey D. Ullman. Equivalences Among Relational Expressions. *SIAM J. Comput.*, 8(2):218–246, 1979. doi:10.1137/0208017.
11. Josh Alman, Timothy M. Chan, and R. Ryan Williams. Polynomial Representations of Threshold Functions and Algorithmic Applications. In *Proceedings of the 2016 IEEE 57th Annual Symposium on Foundations of Computer Science*, FOCS '16, pages 467–476, 2016. doi:10.1109/FOCS.2016.57.
12. Noga Alon, Raphael Yuster, and Uri Zwick. Finding and Counting Given Length Cycles. *Algorithmica*, 17(3):209–223, 1997. doi:10.1007/BF02523189.
13. Arturs Backurs and Piotr Indyk. Edit Distance cannot be computed in strongly subquadratic time (unless SETH is false). In *Proceedings of the 47th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2015, pages 51–58. ACM, 2015. doi:10.1145/2746539.2746612.
14. Arturs Backurs and Piotr Indyk. Which Regular Expression Patterns Are Hard to Match? In *Proceedings of the 2016 IEEE 57th Annual Symposium on Foundations of Computer Science*, FOCS '16, pages 457–466, 2016. doi:10.1109/FOCS.2016.56.
15. Arturs Backurs, Liam Roditty, Gilad Segal, Virginia Vassilevska Williams, and Nicole Wein. Towards tight approximation bounds for graph diameter and eccentricities. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, pages 267–280, 2018. doi:10.1145/3188745.3188950.

16 Guillaume Bagan, Arnaud Durand, and Etienne Grandjean. On Acyclic Conjunctive Queries and Constant Delay Enumeration. In Jacques Duparc and Thomas A. Henzinger, editors, *Computer Science Logic*, pages 208–222. Springer Berlin Heidelberg, 2007.

17 Christoph Berkholz, Jens Keppeler, and Nicole Schweikardt. Answering Conjunctive Queries Under Updates. In *Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, PODS '17, pages 303–318. ACM, 2017. doi:10.1145/3034786.3034789.

18 Karl Bringmann. Why Walking the Dog Takes Time: Fréchet Distance Has No Strongly Subquadratic Algorithms Unless SETH Fails. In *Proceedings of the 2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, FOCS '14, pages 661–670, 2014. doi:10.1109/FOCS.2014.76.

19 Karl Bringmann, Allan Grønlund, and Kasper Green Larsen. A Dichotomy for Regular Expression Membership Testing. In *Proceedings of the 2017 IEEE 58th Annual Symposium on Foundations of Computer Science*, FOCS '17, pages 307–318, 2017. doi:10.1109/FOCS.2017.36.

20 Karl Bringmann and Marvin Künnemann. Quadratic Conditional Lower Bounds for String Problems and Dynamic Time Warping. In *Proceedings of the 2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, FOCS '15, pages 79–97, 2015. doi:10.1109/FOCS.2015.15.

21 Karl Bringmann and Marvin Künnemann. Multivariate Fine-Grained Complexity of Longest Common Subsequence. In *Proceedings of the 29th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '18, pages 1216–1235, 2018. doi:10.1137/1.9781611975031.79.

22 Andrei A. Bulatov. A Dichotomy Theorem for Constraints on a Three-Element Set. In *Proceedings of the 2002 IEEE 43rd Annual Symposium on Foundations of Computer Science*, FOCS '02, pages 649–658, 2002. doi:10.1109/SFCS.2002.1181990.

23 Andrei A. Bulatov. Tractable conservative Constraint Satisfaction Problems. In *18th IEEE Symposium on Logic in Computer Science*, page 321, 2003. doi:10.1109/LICS.2003.1210072.

24 Andrei A. Bulatov. A Dichotomy Theorem for Nonuniform CSPs. In *Proceedings of the 2017 IEEE 58th Annual Symposium on Foundations of Computer Science*, FOCS '17, pages 319–330, 2017. doi:10.1109/FOCS.2017.37.

25 Andrei A. Bulatov. Constraint satisfaction problems: complexity and algorithms. *SIGLOG News*, 5(4):4–24, 2018. doi:10.1145/3292048.3292050.

26 Andrei A. Bulatov, Peter Jeavons, and Andrei A. Krokhin. Classifying the Complexity of Constraints Using Finite Algebras. *SIAM J. Comput.*, 34(3):720–742, 2005. doi:10.1137/S0097539700376676.

27 Nofar Carmeli and Markus Kröll. Enumeration Complexity of Conjunctive Queries with Functional Dependencies. In *ICDT*, volume 98 of *LIPIcs*, pages 11:1–11:17. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018.

28 Marco L. Carmosino, Jiawei Gao, Russell Impagliazzo, Ivan Mihajlin, Ramamohan Paturi, and Stefan Schneider. Nondeterministic Extensions of the Strong Exponential Time Hypothesis and Consequences for Non-reducibility. In *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science*, pages 261–270, 2016. doi:10.1145/2840728.2840746.

29 Timothy M. Chan and Ryan Williams. Deterministic APSP, Orthogonal Vectors, and More: Quickly Derandomizing Razborov-Smolensky. In *Proceedings of the 27th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '16, pages 1246–1255, 2016. URL: http://dl.acm.org/citation.cfm?id=2884435.2884522.

30 Ashok K. Chandra and David Harel. Structure and Complexity of Relational Queries. *J. Comput. Syst. Sci.*, 25(1):99–128, 1982. doi:10.1016/0022-0000(82)90012-5.

31 Ashok K. Chandra and Philip M. Merlin. Optimal Implementation of Conjunctive Queries in Relational Data Bases. In *Proceedings of the 9th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 1977, pages 77–90, 1977. doi:10.1145/800105.803397.

32 Hubie Chen. A rendezvous of logic, complexity, and algebra. *SIGACT News*, 37(4):85–114, 2006. doi:10.1145/1189056.1189076.

33 Nadia Creignou and Miki Hermann. Complexity of Generalized Satisfiability Counting Problems. *Inf. Comput.*, 125(1):1–12, 1996. doi:10.1006/inco.1996.0016.

34 Nadia Creignou, Sanjeev Khanna, and Madhu Sudan. *Complexity classifications of Boolean Constraint Satisfaction problems*. SIAM, 2001. doi:10.1137/1.9780898718546.

35 Víctor Dalmau. Some dichotomy theorems on constant free Boolean formulas. Technical Report LSI-97-43-R, Departament LSI, Universitat Politècnica de Catalunya, 1997.

36 Holger Dell, Marc Roth, and Philip Wellnitz. Counting Answers to Existential Questions. *arXiv e-prints*, February 2019. arXiv:1902.04960.

37 Rodney G. Downey, Michael R. Fellows, and Udayan Taylor. The Parameterized Complexity of Relational Database Queries and an Improved Characterization of W[1]. In *First Conference of the Centre for Discrete Mathematics and Theoretical Computer Science*, pages 194–213, 1996.

38 Tomás Feder and Moshe Y. Vardi. Monotone monadic SNP and constraint satisfaction. In *Proceedings of the 25th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 1993, pages 612–622, 1993. doi:10.1145/167088.167245.

39 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.

40 Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2006. doi:10.1007/3-540-29953-X.

41 Jiawei Gao, Russell Impagliazzo, Antonina Kolokolova, and Ryan Williams. Completeness for First-order Properties on Sparse Structures with Algorithmic Applications. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '17, pages 2162–2181, 2017. URL: http://dl.acm.org/citation.cfm?id=3039686.3039827.

42 Michel Habib, Christophe Paul, and Laurent Viennot. A Synthesis on Partition Refinement: A Useful Routine for Strings, Graphs, Boolean Matrices and Automata. In *STACS 98, 15th Annual Symposium on Theoretical Aspects of Computer Science*, pages 25–38, 1998. doi:10.1007/BFb0028546.

43 Pavol Hell and Jaroslav Nesetril. On the complexity of H-coloring. *J. Comb. Theory, Ser. B*, 48(1):92–110, 1990. doi:10.1016/0095-8956(90)90132-J.

44 Neil Immerman. Upper and Lower Bounds for First Order Expressibility. *J. Comput. Syst. Sci.*, 25(1):76–98, 1982. doi:10.1016/0022-0000(82)90011-3.

45 Russell Impagliazzo and Ramamohan Paturi. On the Complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001. doi:10.1006/jcss.2000.1727.

46 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.

47 Peter Jeavons, David A. Cohen, and Marc Gyssens. Closure properties of constraints. *J. ACM*, 44(4):527–548, 1997. doi:10.1145/263867.263489.

48 Robert Krauthgamer and Ohad Trabelsi. Conditional Lower Bounds for All-Pairs Max-Flow. *ACM Trans. Algorithms*, 14(4):42:1–42:15, 2018. doi:10.1145/3212510.

49 Andrea Lincoln, Virginia Vassilevska Williams, and Ryan Williams. Tight Hardness for Shortest Cycles and Paths in Sparse Graphs. In *Proceedings of the 29th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '18, pages 1236–1252, 2018. URL: http://dl.acm.org/citation.cfm?id=3174304.3175350.

50 Christos H. Papadimitriou and Mihalis Yannakakis. On the Complexity of Database Queries. *J. Comput. Syst. Sci.*, 58(3):407–427, 1999. doi:10.1006/jcss.1999.1626.

51 Mihai Patrascu and Ryan Williams. On the Possibility of Faster SAT Algorithms. In *Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '10, pages 1065–1075, 2010. doi:10.1137/1.9781611973075.86.

52 Liam Roditty and Virginia Vassilevska Williams. Fast approximation algorithms for the diameter and radius of sparse graphs. In *Proceedings of the 45th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2013, pages 515–524, 2013. doi:10.1145/2488608.2488673.

53 Thomas J. Schaefer. The Complexity of Satisfiability Problems. In *Proceedings of the 10th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 1978, pages 216–226, 1978. doi:10.1145/800133.804350.

54 Moshe Y. Vardi. The Complexity of Relational Query Languages (Extended Abstract). In *Proceedings of the 14th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 1982, pages 137–146, 1982. doi:10.1145/800070.802186.

55 Moshe Y. Vardi. On the Complexity of Bounded-Variable Queries. In *Proceedings of the 14th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 266–276, 1995. doi:10.1145/212433.212474.

56 Virginia Vassilevska and Ryan Williams. Finding, Minimizing, and Counting Weighted Subgraphs. In *Proceedings of the 41st Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2009, pages 455–464. ACM, 2009. doi:10.1145/1536414.1536477.

57 Ryan Williams. A New Algorithm for Optimal 2-constraint Satisfaction and Its Implications. *Theor. Comput. Sci.*, 348(2):357–365, December 2005. doi:10.1016/j.tcs.2005.09.023.

58 Ryan Williams. Faster decision of first-order graph properties. In *Joint Meeting of the 23rd EACSL Annual Conference on Computer Science Logic (CSL) and the 29th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 80:1–80:6, 2014. doi:10.1145/2603088.2603121.

59 Ryan Williams. On the Difference Between Closest, Furthest, and Orthogonal Pairs: Nearly-Linear vs Barely-Subquadratic Complexity. In *Proceedings of the 29th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '18, pages 1207–1215, 2018. doi:10.1137/1.9781611975031.78.

60 Dmitriy Zhuk. A Proof of CSP Dichotomy Conjecture. In *Proceedings of the 2017 IEEE 58th Annual Symposium on Foundations of Computer Science*, FOCS '17, pages 331–342, 2017. doi:10.1109/FOCS.2017.38.

Imperfect Gaps in Gap-ETH and PCPs

Mitali Bafna
Harvard University, Cambridge, MA, USA
mitalibafna@g.harvard.edu

Nikhil Vyas
MIT, Cambridge, MA, USA
nikhilv@mit.edu

Abstract

We study the role of perfect completeness in probabilistically checkable proof systems (PCPs) and give a way to transform a PCP with imperfect completeness to one with perfect completeness, when the initial gap is a constant. We show that $\text{PCP}_{c,s}[r,q] \subseteq \text{PCP}_{1,s'}[r+O(1), q+O(r)]$ for $c-s=\Omega(1)$ which in turn implies that one can convert imperfect completeness to perfect in linear-sized PCPs for NP with a $O(\log n)$ additive loss in the query complexity q. We show our result by constructing a "robust circuit" using threshold gates. These results are a gap amplification procedure for PCPs, (when completeness is not 1) analogous to questions studied in parallel repetition [22] and pseudorandomness [15] and might be of independent interest.

We also investigate the time-complexity of approximating perfectly satisfiable instances of 3SAT versus those with imperfect completeness. We show that the Gap-ETH conjecture without perfect completeness is equivalent to Gap-ETH with perfect completeness, i.e. MAX $3\text{SAT}(1-\epsilon, 1-\delta), \delta > \epsilon$ has $2^{o(n)}$ algorithms if and only if MAX $3\text{SAT}(1, 1-\delta)$ has $2^{o(n)}$ algorithms. We also relate the time complexities of these two problems in a more fine-grained way to show that $T_2(n) \leq T_1(n(\log\log n)^{O(1)})$, where $T_1(n), T_2(n)$ denote the randomized time-complexity of approximating MAX 3SAT with perfect and imperfect completeness respectively.

2012 ACM Subject Classification Theory of computation → Problems, reductions and completeness; Theory of computation → Complexity theory and logic

Keywords and phrases PCP, Gap-ETH, Hardness of Approximation

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.32

Funding *Mitali Bafna*: Supported by NSF Grant CCF-1565641 and CCF-1715187.
Nikhil Vyas: Supported by NSF Grant CCF-1741615.

Acknowledgements We would like to thank our advisors Madhu Sudan and Ryan Williams for helpful discussions. We are also grateful to all the reviewers, for detailed comments on the paper.

1 Introduction

The PCP theorem [3] was a breakthrough result that proved that NP has proofs that can be verified using just $O(1)$ bits and constant probability of error, with a minimal blow-up in the size of the proof. The theorem led to a flurry of activity in getting the best set of parameters: the soundness, proof size and queries. These PCP constructions were instrumental in showing optimal hardness of approximation results for a host of problems like k-SAT and 3LIN [18]. Despite this progress, many important questions remain wide open, for instance: Do there exist linear-sized PCPs for NP, with constant queries and constant soundness? Hence we believe it is important to understand the role of all the parameters in PCPs for NP and we focus our attention on the completeness of these proof systems.

We investigate the question that can imperfect completeness help to get better PCPs? The size versus query tradeoff in PCPs has been extensively studied: A long line of work culminated in a PCP [11] with $O(n\text{polylog } n)$ size and $O(1)$ queries. On the other hand, Ben-Sasson et al [6] achieved a linear-sized PCP with $O(n^{\epsilon})$ query size for all constants

$\epsilon > 0$.[1] These results are far from the conjectured $O(1)$ queries and linear-size. We show that one can transform any PCP with imperfect completeness and constant gap (between soundness and completeness) to one with perfect completeness with a mild additive loss in the number of queries. The loss in query complexity that we get in the transformation from imperfect to perfect completeness in the latter regime (of linear-size) is inconsequential in comparison to the query complexity of [6].

Although in current PCP constructions perfect completeness might come for free when one does not care about the verifier's predicate, PCPs with imperfect completeness are very important in showing optimal hardness of approximation for problems like 3LIN [18], where deciding satisfiability is in polytime. For other CSPs like Max 1-in-k-SAT one can get substantially better approximation algorithms for perfectly satisfiable instances [17]. The infamous Unique games conjecture of Khot [20] asks for a PCP with unique queries and imperfect completeness, the latter being necessary due to the tractability of satisfiable instances of Unique games. Although the imperfect completeness in the previous cases was necessary, in the case of CSPs like 2-to-1 games and Max k-CSP one would guess that the same hardness of approximation results should hold with perfect completeness. Unfortunately all the current methods [13, 10] incur a loss in completeness and it is unclear whether this is because of the nature of the problem or due to the inefficacy of current methods. This leads to the central question that given a CSP, how hard is it to approximate instances that are perfectly satisfiable as compared to those that are not?

We also study this question in a fine-grained way and compare the time complexities of approximating satisfiable versus imperfectly satisfiable instances of 3SAT. NP-hardness results while very useful in measuring intractability with respect to poly-time algorithms, do not imply tight or even superpolynomial lower bounds for the running time. The Exponential Time Hypothesis (ETH) [19] states that there are no $2^{o(n)}$ time algorithms for deciding satisfiability of 3SAT. Through the equivalence between PCPs and gap problems, using state of the art PCPs [11, 7] there is a reduction from a 3SAT instance on n variables to a Gap-3SAT instance with $O(n\text{polylog } n)$ variables. This proves that under ETH, Gap-3SAT does not have $O(2^{n/\log^c(n)})$ algorithms for some fixed c, whereas Gap-3SAT has eluded even $2^{o(n)}$ algorithms. To get around precisely this gap, the Gap-ETH hypothesis was proposed [12, 21]. Gap-ETH states that Gap-3SAT does not have $2^{o(n)}$ algorithms. This hypothesis has led to several tight inapproximability results [9, 14, 8, 1] with respect to the running time required. We study the role of perfect completeness in Gap-ETH, where Gap-ETH without perfect completeness is the hypothesis that there are no $2^{o(n)}$ algorithms for Gap-3SAT without perfect completeness.

Gap amplification is in itself an important problem studied in the context of parallel repetition [23], error reduction and pseudorandomness [15]. We study this problem in PCPs and show a way to transform any PCP into a one-sided error one. Similar questions of gap amplification when completeness is not 1, have been studied for parallel repetition [22], but these results incur a huge blow-up in the alphabet, which soon becomes $\Omega(1)$ and cannot be applied to get perfect completeness in PCPs. These techniques in parallel repetition have been used in quantum computation, to show instances of multi-player games with large separation between the entangled and classical value and amplification of entangled games [22, 4].

[1] This particular construction is non-uniform. To our knowledge no explicit PCPs with $o(n)$ query complexity, constant soundness and linear size are known.

1.1 Our contributions

PCPs without perfect completeness

We show a way to boost the completeness of PCPs which makes the completeness 1. Our results go via the construction of "robust circuits" for the approximate threshold function on n bits. These circuits are of depth $O(\log n)$, fan-in $O(1)$ and size $O(n)$, and use successive layers of threshold gates to boost the fraction of ones in inputs that have large Hamming weight, while maintaining the fraction of ones in other inputs below a certain threshold. The circuits are tolerant to some form of adversarial corruptions and this property allows us to prove the soundness of the new PCP. Our main theorem is the following:

▶ **Theorem 1.** *Let $c, s \in (0,1), s < c$ be constants then there exists a constant $s' \in (0,1)$ depending only on c, s such that,*

$$PCP_{c,s}[r,q] \subseteq PCP_{1,s'}[r, q + O_{s,c}(r)]$$

furthermore if the original proof size was n then the final proof size in $n + O(2^r)$.

Note that in the above theorem, one can prove inclusion in a PCP class, with arbitrary constant s'' (instead of a fixed constant s') by applying derandomized serial repetition ($\text{PCP}_{1,s'}[r,q] \subseteq \text{PCP}_{1,s''}[r, O(q)]$ with same proof size). This does not blow up the size of the PCP and the query complexity only increases by a constant factor.

As a corollary we show that linear-sized PCPs for NP with imperfect completeness, can be converted to a linear-sized PCPs with perfect completeness and $q + O(\log n)$ queries. Current PCP constructions with constant rate and alphabet have query complexity $n^{\Omega(1)}$[6], so we show that for improving upon this, it is enough to get linear sized PCPs with imperfect completeness and better query complexity.

We also consider the notion of "randomized reduction between PCPs", defined below. Bellare et al [5] considered the notion of a randomized reduction R between two promise problems given by sets (A_1, B_1) and (A_2, B_2). A randomized polynomial time reduction R from promise problems $(A_1, B_1) \leq_R (A_2, B_2)$ with error probability p satisfies:

1. if $x \in A_1$ then w.p. $\geq 1 - p$, $R(x) \in A_2$.
2. if $x \in B_1$ then w.p. $\geq 1 - p$, $R(x) \in B_2$.

This notion naturally extends to PCP complexity classes. We give a randomized reduction between PCP classes with imperfect and perfect completeness.

▶ **Theorem 2.** *Let $c, s \in (0,1), s < c$ be constants then there exists a constant $s' \in (0,1)$ depending only on c, s such that,*

$$PCP_{c,s}[r,q] \leq_R PCP_{1,s'}[r, q + O_{s,c}(\log r)]$$

with probability $1 - 2^{-\Omega(r)}$. Furthermore if the original proof size was n then the final proof size in $n + O(2^r)$.

Gap-ETH without perfect completeness

We study the relation between time complexities of approximating satisfiable instances of MAX 3SAT versus that of approximating unsatisfiable instances. We first show the equivalence of the Gap-ETH conjecture with perfect and imperfect completeness. We formally state the Gap-ETH conjecture below:

▶ **Conjecture 1** (Gap Exponential-Time Hypothesis (Gap-ETH) [12, 21]). *For some constants $\delta, \epsilon > 0$, no algorithm can, given a 3-SAT formula ϕ on n variables and $m = O(n)$ clauses, solve the decision problem MAX 3-SAT$(1, 1-\epsilon)$ in $O(2^{\delta n})$ time.*

There are many versions of the Gap-ETH conjecture that one can consider. Many works study the randomized Gap-ETH conjecture which says that there are not even any randomized algorithms that can decide Max 3-SAT$(1, 1-\epsilon)$. We show the following theorem:

▶ **Theorem 3.** *If there exists a randomized (with no false positives) $2^{o(n)}$ time algorithm for MAX 3SAT$(1, 1-\gamma)$ for all constant $\gamma > 0$ then there exists a randomized(with no false positives) $2^{o(n)}$ time algorithm for MAX 3SAT$(s(1+\epsilon), s)$ for all constants $s, \epsilon > 0$.*

Algorithms with no false positives are interesting as a) Randomized SAT (not MAX-SAT) algorithms can be modified to have no false positives by self-reduction b) some of the hardness results from Gap-ETH go through reductions which do not produce false positives [8]. This allows us to compose without losing in hardness by assuming Gap-ETH with one sided error. This was also the notion considered by Applebaum [2] to give hardness of Gap-ETH. As the original Gap-ETH hypothesis [12, 21] talks about deterministic algorithms we would prefer to get a deterministic reduction between these two problems.

We can get more fine-grained results relating the time-complexities of MAX 3SAT with perfect and imperfect completeness using the Theorem 2 stated earlier.

▶ **Corollary 4.** *If there exists a $T(n)$ time algorithm for MAX 3SAT$(1, 1-\delta)$ for all $\delta > 0$ then there exists a $T(n(\log\log n)^{O(1)})$ time randomized algorithm for MAX 3SAT$(1-\epsilon, 1-\gamma)$ for all $\epsilon, \gamma, 0 < \epsilon < \gamma$.*

1.2 Previous work

Bellare et al [5] also studied the problem transforming probabilistically checkable proofs with imperfect completeness to those with perfect completeness. Their techniques do not yield any inclusions for PCP classes. They proved the following randomized reduction between PCP classes:

$$\text{PCP}_{c,s}[r, q] \leq_R \text{PCP}_{1,rs/c}[r, qr/c]$$

For constant c and $r = \omega(1)$, they lose a multiplicative factor of r in the soundness, which makes the theorem non-trivial only when $s = o(1)$.

2 Preliminaries

Throughout the paper we follow this notation:

Notation

$\text{Thr}_\delta(x_{i_1}, \ldots, x_{i_r})$ = threshold at δ-fraction taken on the set of bits $\{x_{i_1}, \ldots, x_{i_r}\}$. We also use $\text{Thr}_\delta(x|_S)$ to mean that the threshold is with respect to the bits of x restricted to $S \subseteq [n]$ and sometimes drop the x and δ to use $\text{Thr}(S)$, when the input/fraction being used is clear from context. $exp(x)$ refers to e^x. For a string $x \in \{0,1\}^n$, let $\bar{x} = \frac{1}{n}\sum_i x_i$, denote the average number of 1's in x.

MAX k-CSP(c, s) - the promise problem of deciding whether there exists an assignment satisfying more than c-fraction clauses or every assignment satisfies at most s fraction of clauses. When the CSP is a 3SAT instance, it is denoted by MAX 3SAT(c, s).

Firstly we discuss some standard probability bounds like the Chernoff bound and the Lovász local lemma.

2.1 Chernoff Bounds

1. Multiplicative Chernoff bound 1: Let $X = \frac{1}{n}X_i$, where $X_1, \ldots, X_n$ are random variables in $\{0,1\}$, with $E[X] = \mu$. Then for all $\delta \geq 1$,

   $$\Pr[X > (1+\delta)\mu] \leq exp(-\Omega(\delta\mu))$$

   for $\delta \leq 1$,

   $$\Pr[X > (1+\delta)\mu] \leq exp(-\delta^2\mu/3))$$

   $$\Pr[X < (1-\delta)\mu] \leq exp(-\delta^2\mu/2))$$

2. Multiplicative Chernoff bound 2: Let $X = \frac{1}{n}X_i$, where $X_1, \ldots, X_n$ are random variables in $\{0,1\}$, with $E[X] = \mu$. Then for all $\delta \geq 2$,

   $$\Pr[X > (1+\delta)\mu] \leq \left(\frac{e^\delta}{(1+\delta)^{1+\delta}}\right)^\mu \leq \exp(-\Omega(\delta(\log(1/\delta))\mu)).$$

▶ **Lemma 5** (Lovász local lemma). *Let $E_1, E_2, \ldots, E_n$ with $\Pr[E_i] = 1$ be events such that any E_i is independent of all but d other events. Then if $pe(d+1) \leq 1$ then*

$$\Pr\left[\bigcap E_i\right] \geq (1-1/d)^n$$

Let us now define probabilistic proof systems. Firstly, we define the notion of an (r,q)-restricted verifier: For integer valued functions $r(\cdot)$ and $q(\cdot)$, a verifier is said to be (r,q)-restricted if on every input of length n, it tosses at most $r(n)$ coins and queries the proof in at most $q(n)$ bits non-adaptively.

▶ **Definition 6** (PCP). *For integer-valued functions $r(\cdot), q(\cdot)$ and functions $c(\cdot), s(\cdot)$ mapping to $[0,1]$, the class $PCP_{c,s}[r,q]$ consists of all languages for which there exists an (r,q)-restricted non-adaptive verifier V with the following properties:*

1. *Completeness: For all $x \in L$, there exists a proof π such that $V^\pi(x)$ accepts with probability at least c (over the coin tosses of V).*
2. *Soundness: For all $x \notin L$, for all proofs π, $V^\pi(x)$ accepts with probability at most s.*

We now go to the notion of averaging samplers. Averaging samplers are used to derandomize the process of random sampling to estimate the average number of ones in a string $x = \{0,1\}^n$, see survey of [16]. We use the following sampler therein:

▶ **Lemma 7.** *The expander sampler with parameters (δ, ϵ, N) is an expander graph on N vertices, such that the neighbors of a vertex i, specify a sample $S_i \subseteq [N]$. The set family satisfies the following properties:*

1. *For all i, $|S_i| = \frac{1}{\delta\epsilon^2}$*
2. *For every S_i the number of sets S_j which intersect with it are $O\left(\frac{1}{\delta^2\epsilon^4}\right)$.*
3. *For any string $x \in \{0,1\}^N$, $\Pr_{S \sim \mathcal{ES}(\delta,\epsilon,N)}[|\overline{(x|_S)} - \overline{x}| > \epsilon] \leq \delta$, where $\overline{(x|_S)}$ denotes the average of x taken over the positions specified by S.*

We analyse the standard expander sampler given above and prove that one can get a sampler with the following properties. In the appendix, we provide a detailed proof.

▶ **Theorem 8** (Sampler). *For all constants ϵ, δ, γ, there exists a constant C such that, there is a set family $\mathcal{S}(\epsilon, \delta, \gamma, N) = (S_i)_{i=1}^{N/2}$ on $[N]$ with the following properties:*

1. *For any string* $x \in \{0,1\}^N$, $\Pr_{S\sim\mathcal{S}}[|\overline{(x|_S)} - \overline{x}| > \epsilon] \leq \delta$.
2. *For all* $\eta < (1-\gamma)/2$, *for any string* $x \in \{0,1\}^N$, *where* $\overline{x} \geq 1-\eta$, *we get that,* $\Pr_{S\sim\mathcal{S}}[\overline{(x|_S)} < \gamma] \leq \eta/2$.
3. *For all* i, $|S_i| = C = O_{\epsilon,\delta,\gamma}(1)$.
4. *The number of sets in* $\mathcal{S}$ *is* $N/2$.

3 PCPs without perfect completeness

In this section we prove that PCPs with imperfect completeness can be converted to ones with perfect completeness with a mild blow-up in queries.

3.1 Reductions with minimal Query Blow-up

We first show a reduction that preserves the randomness complexity while losing an additive factor in the queries.

▶ **Reminder of Theorem 1.** *For all constants* $c, s \in (0,1), s < c$, *there exists a constant* $s' \in (0,1)$, *such that for all integer-valued functions* $r(\cdot), q(\cdot)$, *the following is true:*

$$PCP_{c,s}[r,q] \subseteq PCP_{1,s'}[r, q + O_{s,c}(r)].$$

Furthermore if the original proof size was n, *then the final proof size will be* $n + O(2^r)$.

For notational simplicity we will prove that:

$$\mathrm{PCP}_{9/10,6/10}[r,q] \subseteq \mathrm{PCP}_{1,9/10}[r, q + O(r)],$$

with proof size $n + O(2^r)$. All constants that follow are universal constants, although in full generality, they only depend on c, s that we have fixed to $(9/10, 6/10)$.

The rest of this section is devoted to the proof of this theorem. The main idea here is to build a "robust circuit" of small depth, using threshold gates of small fan-in, over the proof oracle of the original PCP. We then ask the new prover to provide the original proof and along with that, also ask for what each gate in the circuit evaluates to, when provided the original clause evaluations as input. As discussed earlier, the circuit boosts the fraction of ones in every layer, for inputs x that satisfy $\overline{x} \geq 9/10$, while maintaining the fraction of ones for inputs that satisfy $\overline{x} \leq 7/10$. We need to do this boosting step by step so that the fan-in does not blow up, and also need to use threshold gates that take "random" subsets of inputs from the previous layer, so that the ones in the input get distributed across all the gates. We get rid of the random subsets, by using any standard sampler over the gates of the previous layer.

Let us now describe the circuit more formally. Later we will give a way to get complete PCPs from incomplete ones using this circuit.

Description of Circuit $\Gamma_m(\cdot)$:

- The circuit has $d = \log m$ layers, $L_1, \ldots, L_d$, with layer i composed of $w_i = m/2^i$ gates denoted by $L_{i1}, \ldots, L_{iw_i}$. The zeroth layer L_0 is the m inputs to the circuit.
- Every gate $L_{(i+1)j}$ is a threshold gate $\mathrm{Thr}_{0.8}$. Let the set family given by the sampler from Theorem 8 on w_i nodes with parameters $\mathcal{S}(1/10, 6/10, 8/10, w_i) = (S_{(i+1)j})_{j=1}^{w_{i+1}}$. Let $L_{(i+1)j} = \mathrm{Thr}_{0.8}(L_i|_{S_{(i+1)j}})$. By property 3 of expander sampler fan-in $= |S_{(i+1)j}| = O(1)$.

We now use this circuit to give our main reduction.

Proof of Theorem 1. Let $L \subseteq \{0,1\}^*$ be a language in $\text{PCP}_{9/10,6/10}[r,q]$ via the proof system $\mathcal{P} = (\Pi, Q)$, where Π and Q denote the proof and the set of queries. We can now use the equivalence between MAX q-CSP(c,s) and PCPs, to get a set of clauses $\mathcal{C} = \{C_1, \ldots, C_m\}$ of width q, for $m = 2^r$, such that $L \leq$ MAX q-$\mathcal{C}(9/10, 6/10)$. (When $y \in L$, then there exists an assignment x, such that 9/10-fraction of the clauses when evaluated on x output 1, whereas when $y \notin L$, for every assignment x, at most 6/10 of the clauses evaluate to 1.)

To prove the theorem, we will give a new proof system $\mathcal{P}' = (\Pi', Q')$ for L, that has perfect completeness and soundness equal to 9/10. We will transform $\mathcal{P}$ using the circuit $\Gamma_m(\cdot)$ described above, to get $\mathcal{P}'$. We consider the circuit $\Gamma_m(C_1(\Pi), \ldots, C_m(\Pi))$ and ask the new prover to give one bit for every gate of the circuit. More precisely, we ask the new prover to give bits of Π (interpreted as an assignment $x \in \{0,1\}^n$ for the MAX q-CSP: $\mathcal{C}$) and in addition gives bits for every layer in the circuit Γ_m:

$$\ell_i = \{\ell_{i1}, \ldots, \ell_{iw_i}\}, \forall i \in \{0, 1, \ldots, d\}.$$

These bits are supposed to correspond to a correct evaluation of the circuit Γ_m when given $(C_1(x), \ldots, C_m(x))$ $(\Pi = x)$ as input. That is, ideally the prover should give us, $\ell_{0j} = C_j(x), \forall j \in [m]$ and $\ell_{(i+1)j} = L_{(i+1)j}(\ell_i), \forall i \in [d], j \in w_i$, where $L_{(i+1)j}(\ell_i)$ denotes the gate $L_{(i+1)j}$ evaluated on the output bit vector ℓ_i of the previous layer. We probabilistically test this using a new set of queries Q', described below.

Verifier Checks (Q'). For notational simplicity in describing the queries of the new verifier, we will do the following. For each layer i (that has $m/2^i$ gates), consider 2^i copies of the set of gates L_i, and let this new set be denoted by $L'_{i1}, \ldots, L'_{im}$ with corresponding proof bits by $\ell'_i = \{\ell'_{i1}, \ldots, \ell'_{im}\}$ and each gate having its set of inputs $(S'_{i1}, \ldots, S'_{im})$. Note that this duplication of bits/gates is only for description of the queries, and the prover will only give $m/2^i$ bits for every layer i.

Intuitively, we will check whether every gate is correct with respect to its immediate inputs (from the layer below it) and whether the final gate (on the topmost layer) evaluates to 1. To do so, the verifier tosses $\log m$ random coins and on random string $j \in [m]$, it checks whether the following is true:

$$Q'_j := (C_j(x) \stackrel{?}{=} \ell'_{0j}) \wedge (L'_{1j}(\ell_0) \stackrel{?}{=} \ell'_{1j}) \ldots \wedge (L'_{dj}(\ell_{d-1}) \stackrel{?}{=} \ell'_{dj}) \wedge \ell'_{dj},$$

where the clause $(L'_{ij}(\ell_{i-1}) \stackrel{?}{=} \ell'_{ij})$ outputs 1 iff $(L'_{ij}(\ell_{i-1})$ equals $\ell'_{ij})$. As explained earlier, each of the clauses, checks whether the gate L'_{ij} is correct, with respect to its input layer $\ell_{(i-1)}$. Notice here that each check Q_j, checks one gate in every layer and furthermore these checks are uniform across a layer, i.e. every gate in a layer is checked with the same probability.

To perform the check above, we query the proof bits $\ell_{i-1}|_{S'_{ij}}$, making a constant number of queries, since the fanin of every gate is a fixed constant, i.e. L_{ij} has fanin $|S'_{ij}| = O(1)$. We then evaluate the threshold gate L_{ij} on these bits and take the $\wedge$ across the layers. The check $(C_j(x) \stackrel{?}{=} \ell'_{0j})$ needs to query q queries to x, hence the total number of queried proof bits is $q + O(\log m) = q + O(r)$. Further note that the randomness complexity of the verifier remains the same as before i.e. $= r = \log m$.

We now prove the completeness and soundness of the protocol $\mathcal{P}'$.

Completeness. If the original proof system $\mathcal{P}$ had completeness 9/10, then there exists a proof $\Pi = x$ which satisfies 9/10 of the clauses $\mathcal{C}$. The new prover can give us the bit vectors, x and in addition the evaluations of the circuit $\Gamma(C_1(x), \ldots, C_m(x))$, i.e. $x, \ell_0 := (C_j(x))_{j=1}^m$ and $\ell_i := (L_{ij}(\ell_{i-1}))_{j=1}^m$.

In Lemma 9, we prove that, $\overline{\ell_i} \geq 1 - \frac{2^{-i}}{10}$. Since $d = \log m$ and the number of gates on level d is $O(1)$, we get that the fraction of 1s in z_d is $\geq 1 - 1/m$, which gets rounded to 1, since there is only one gate in the topmost layer. Since every query Q'_j checks the consistency of a set of gates and if the bit $\ell_{dj} = 1$, we get completeness equals 1.

Soundness. If the original proof system $\mathcal{P}$ had soundness 6/10, then for all proofs Π that the prover might give, Π satisfies $\leq 6/10$ of the clauses $\mathcal{C}$. Let $\Pi' = (x, \ell_0, \ldots, \ell_d)$ be the proof provided by the new prover.

Let $z_0 := (C_j(x))_{j=1}^m$ and $z_{i+1} := (L_{(i+1)j}(\ell_i))_{i=1}^{w_i}$ be the true local evaluations. Note here that, z_{i+1} is the evaluation bits of layer L_{i+1} evaluated on the bits that the prover provides in the previous layer, ℓ_i. By the soundness of $\mathcal{P}$ we get that x satisfies at most 6/10 of $\mathcal{C}$ which means that $\overline{z_0} \leq 6/10$.

Now we have two cases:

1. The prover provided the bit vectors ℓ_i such that they agree with the true evaluations z_i in most places, i.e.

$$\forall i, \Pr_{j \sim [w_i]}[\ell_{ij} \neq z_{ij}] \leq 1/10.$$

   Hence we have that $\overline{\ell_0} \leq \overline{z_0} + 1/10 \leq 7/10$. Lemma 10 gives us that for $\overline{\ell_i} \leq 7/10$, $\overline{L_{i+1}(\ell_i)} \leq 6/10$ and therefore $\overline{z_{i+1}} \leq 6/10$. Hence we get that by induction, for all i, $\overline{z_i} \leq 6/10$ and $\overline{\ell_i} \leq 7/10$, and more importantly $\overline{\ell_d} \leq 7/10$. Recall that our verifier checks are uniform over the every layer, and since $\ell_{dj} = 1$ is required for verifier's j^{th} check, Q_j to succeed, we get that soundness is $\leq 7/10$.
2. There exists a layer $i \in \{0, \ldots, d\}$ such that:

$$\Pr_{j \sim [w_i]}[\ell_{ij} \neq z_{ij}] > 1/10.$$

   Since z_{ij}'s are the correct evaluations, the above implies that, the prover's proof will fail the local checks in 1/10-fraction of the gates of layer i. Since the verifier checks are uniform over the gates of every layer, (i.e. they check the gate of each layer with the same probability), the verifier checks the incorrect gates with probability at least 1/10. Hence the soundness in this case is $\leq 9/10$.

Note that one of these cases has to occur, hence the overall soundness is the maximum of the two cases, i.e. $\leq 9/10$.

Proof Length. Every layer L_i has width $m/2^i$. Thus the total number of gates in the circuit is $m + m/2 + \ldots = O(m) = O(2^r)$. Since Π' consists of the original proof appended with the circuit evaluations, the proof length is $n + O(2^r)$. ◀

We now complete the proofs of completeness and soundness in Theorem 1.

▶ **Lemma 9** (Completeness). *Let $y_0 \in \{0,1\}^m$ be such that $\overline{y_0} \geq 9/10$. Let $y_i \in \{0,1\}^{w_i}$ denote the output string of layer i, when $\mathcal{C}$ is evaluated with the zeroth layer set to y_0. Then we have that for all i, y_i satisfies $\overline{y_i} \geq 1 - \frac{2^{-i}}{10}$.*

Proof. We will prove the lemma by induction on i. Note that the base case $i = 0$, holds trivially. Now consider the $(i+1)^{th}$ layer of the circuit and the gates $L_{(i+1)j}$ that take as input the set $S_{(i+1)j}$ corresponding to the expander sampler on w_i bits. By the induction hypothesis we have that y_i is such that $\overline{y_i} \geq 1 - \frac{2^{-i}}{10}$. By the expander sampler property 2 with parameters $(1/10, 6/10, 8/10, w_i)$ we get that,

$$\Pr_{j \sim [w_{i+1}]}[L_{(i+1)j}(y_i) = \mathrm{Thr}(y_i|_{S_{(i+1)j}}) = 0] \leq \Pr_{j \sim [w_{i+1}]}[\overline{(y_i|_{S_{(i+1)j}})} < 0.8] \leq \left(\frac{1}{2}\right)\left(\frac{2^{-i}}{10}\right).$$

Which directly implies

$$\Pr_{j\sim[w_{i+1}]}[L_{(i+1)j}(y_i)=1] \geq 1-\frac{2^{-i-1}}{10} \Leftrightarrow \overline{y_{i+1}} \geq 1-\frac{2^{-i-1}}{10}$$

which completes the induction. ◀

▶ **Lemma 10** (Soundness). *Let $y_i \in \{0,1\}^{w_i}$ denote an instantiation of the output gates of layer i with $\overline{y_i} \leq 7/10$. Let $y_{i+1} = L_{i+1}(y_i)$ denote the output of layer $i+1$ when evaluated on the string y_i. Then we have that y_{i+1} satisfies $\overline{y_{i+1}} \leq 6/10$.*

Proof. Recall that in the circuit, the gate $L_{(i+1)j}$ took as input the set $S_{(i+1)j}$ corresponding to the sampler on w_i bits. By the expander sampler property 1, with parameters $(1/10, 6/10, 8/10, w_i)$ we get that, for any string $y_i \in \{0,1\}^{w_i}$ with $\overline{y_i} \leq 7/10$:

$$\Pr_{j\sim[w_{i+1}]}[|\overline{(y_i|_{S_{(i+1)j}})} - 7/10| > 1/10] \leq \Pr_{j\sim[w_{i+1}]}[L_{(i+1)j}(y_i) = \mathrm{Thr}(y_i|_{S_{(i+1)j}}) = 1] \leq 6/10$$

which directly implies $\overline{y_{i+1}} \leq 6/10$ completing the proof.

◀

Theorem 1 implies the following transformation from linear sized PCPs with imperfect completeness to linear sized PCPs with perfect completeness.

▶ **Corollary 11.** *If* NP $\subseteq PCP_{9/10,6/10}[\log n + O(1), q]$ *then* NP $\subseteq PCP_{1,9/10}[\log n + O(1), q + O(\log n)]$.

4 Randomized reductions between PCPs

In this section we prove that PCPs with imperfect completeness can be reduced using randomness to ones with perfect completeness with a lesser blow-up in queries compared to Section 3. We construct a circuit similar to the one in the previous section, but this time we use a randomized circuit to get better parameters and show that our reduction works with high probability. This is our main theorem:

▶ **Reminder of Theorem 2.** *For all constants $c, s \in (0,1), s < c$, there exists a constant $s' \in (0,1)$, such that for all integer-valued functions $r(\cdot), q(\cdot)$, the following is true:*

$PCP_{c,s}[r,q] \leq_R PCP_{1,s'}[r, q + O_{s,c}(\log r)]$.

Furthermore if the original proof size was n, then the final proof size will be $n + O(2^r)$.

For notational simplicity we will prove that:

$\mathrm{PCP}_{9/10,6/10}[r,q] \leq_R \mathrm{PCP}_{1,9/10}[r, q + O(\log r)]$,

with proof size $n + O(2^r)$. All constants that follow are universal constants, although in full generality, they only depend on c, s that we have fixed to $(9/10, 6/10)$.

This immediately implies the following corollary using the query reduction[2] result by Dinur [11],

[2] This result reduces queries to a constant but blows-up the proof size.

▶ **Corollary 12.** *If there exists a $T(n)$ time algorithm for MAX 3SAT$(1, 1-\delta)$ for all $\delta > 0$ then there exists a $T(n(\log\log n)^{O(1)})$ time randomized algorithm for MAX 3SAT$(1-\epsilon, 1-\gamma)$ for all $\epsilon, \gamma, 0 < \epsilon < \gamma$.*

The rest of this section is devoted to the proof of theorem 2. The main idea as in Theorem 1 is to build a "robust circuit" of small depth, using threshold gates of small fan-in, over the proof oracle of the original PCP. We then ask the new prover to provide the original proof and along with that, also ask for what each gate in the circuit evaluates to, when provided the original clause evaluations as input. As discussed earlier, the circuit boosts the fraction of ones in every layer, for inputs x that satisfy $\overline{x} \geq 9/10$, while maintaining the fraction of ones for inputs that satisfy $\overline{x} \leq 7/10$. We need to do this boosting step by step so that the fan-in does not blow up, and also need to use threshold gates that take random subsets of inputs from the previous layer, so that the ones in the input get distributed across all the gates.

Let us now describe the circuit more formally. Later we will give a way to get complete PCPs from incomplete ones using this circuit.

Description of Circuit $\Gamma_m(\cdot)$:

- The circuit has $d = \log\log m$ layers, $L_1, \ldots, L_d$, with layer i composed of $w_i = m/2^i$ gates denoted by $L_{i1}, \ldots, L_{iw_i}$. The zeroth layer L_0 is the m inputs to the circuit.
- Every gate L_{ij} is a threshold gate $\text{Thr}_{0.8}$. A gate L_{ij} takes as inputs a random set of f gates from the previous layer L_{i-1}, i.e. we pick a uniformly random set S_{ij} of size f, (sampled with replacement) from $[m/2^{i-1}]$ and connect gate L_{ij} with gates $L_{(i-1)k}, \forall k \in S_{ij}$.

We now use this circuit to give our main reduction.

Proof of Theorem 2. Let $L \subseteq \{0,1\}^*$ be a language in $\text{PCP}_{9/10,6/10}[r, q]$ via the proof system $\mathcal{P} = (\Pi, Q)$, where Π and Q denote the proof and the set of queries. We can now use the equivalence between MAX q-CSP(c, s) and PCPs to get a set of clauses $\mathcal{C} = \{C_1, \ldots, C_m\}$ of width q, for $m = 2^r$, such that $L \leq$ MAX q-$\mathcal{C}(9/10, 6/10)$. (When $y \in L$, then there exists an assignment x, such that $9/10$-fraction of the clauses when evaluated on x output 1, whereas when $y \notin L$, for every assignment x, at most $6/10$ of the clauses evaluate to 1.)

To prove the theorem, we will give a new proof system $\mathcal{P}' = (\Pi', Q')$ for L, that has perfect completeness and soundness equal to $9/10$. We will transform $\mathcal{P}$ using the circuit $\Gamma_m(\cdot)$ described above, to get $\mathcal{P}'$. We consider the circuit $\Gamma_m(C_1(\Pi), \ldots, C_m(\Pi))$ and ask the new prover to give one bit for every gate of the circuit. More precisely, we ask the new prover to give bits of Π (interpreted as an assignment $x \in \{0,1\}^n$ for the MAX q-CSP: $\mathcal{C}$) and in addition gives bits for every layer in the circuit Γ_m:

$$\ell_i = \{\ell_{i1}, \ldots, \ell_{iw_i}\}, \forall i \in \{0, 1, \ldots, d\}.$$

These bits are supposed to correspond to a correct evaluation of the circuit Γ_m when given $(C_1(x), \ldots, C_m(x))$ $(\Pi = x)$ as input. That is, ideally the prover should give us, $\ell_{0j} = C_j(x), \forall j \in [m]$ and $\ell_{(i+1)j} = L_{(i+1)j}(\ell_i), \forall i \in [d], j \in w_i$, where $L_{(i+1)j}(\ell_i)$ denotes the gate $L_{(i+1)j}$ evaluated on the output bit vector ℓ_i of the previous layer. We probabilistically test this using a new set of queries Q', described below.

Verifier Checks (Q'). For notational simplicity in describing the queries of the new verifier, we will do the following. For each layer i (that has $m/2^i$ gates), consider 2^i copies of the set of gates L_i, and let this new set be denoted by $L'_{i1}, \ldots, L'_{im}$ with corresponding proof bits by $\ell'_i = \{\ell'_{i1}, \ldots, \ell'_{im}\}$ and each gate having its set of inputs $(S'_{i1}, \ldots, S'_{im})$. Note

that this duplication of bits/gates is only for description of the queries, and the prover will only give $m/2^i$ bits for every layer i.

Intuitively, we will check whether every gate is correct with respect to its immediate inputs (from the layer below it) and whether the final gate (on the topmost layer) evaluates to 1. To do so, the verifier tosses $\log m$ random coins and on random string $j \in [m]$, it checks whether the following is true:

$$Q'_j := (C_j(x) \stackrel{?}{=} \ell'_{0j}) \wedge (L'_{1j}(\ell_0) \stackrel{?}{=} \ell'_{1j}) \ldots \wedge (L'_{dj}(\ell_{d-1}) \stackrel{?}{=} \ell'_{dj}) \wedge \ell'_{dj},$$

where the clause $(L'_{ij}(\ell_{i-1}) \stackrel{?}{=} \ell'_{ij})$ outputs 1 iff $(L'_{ij}(\ell_{i-1})$ equals $\ell'_{ij})$. As explained earlier, each of the clauses, checks whether the gate L'_{ij} is correct, with respect to its input layer $\ell_{(i-1)}$. Notice here that each check Q_j, checks one gate in every layer and furthermore these checks are uniform across a layer, i.e. every gate in a layer is checked with the same probability.

To perform the check above, we query the proof bits $\ell_{i-1}|_{S'_{ij}}$, making a constant number of queries, since the fanin of every gate is a fixed constant, i.e. L_{ij} has fanin $|S'_{ij}| = O(1)$. We then evaluate the threshold gate L_{ij} on these bits and take the $\wedge$ across the layers. The check $(C_j(x) \stackrel{?}{=} \ell'_{0j})$ needs to query q queries to x, hence the total number of queried proof bits is $q + O(\log \log m) = q + O(\log r)$. Further note that the randomness complexity of the verifier remains the same as before, $= r = \log m$.

We now prove the completeness and soundness of the protocol $\mathcal{P}'$. Since the reduction is randomized, this boils down to proving that, 1) Completeness: given a Max q-CSP that was c-satisfiable, with high probability it gets mapped to a Max q'-CSP that is perfectly satisfiable and 2) Soundness: given a Max q-CSP that was at most s-satisfiable, with high probability it gets mapped to a Max q'-CSP that is at most s'-satisfiable.

Completeness. If the original proof system $\mathcal{P}$ had completeness 9/10, then there exists a proof $\Pi = x$ which satisfies 9/10 of the clauses $\mathcal{C}$. The new prover can give us the bit vectors, x and in addition the evaluations of the circuit $\Gamma(x)$, i.e. $x, \ell_1 := (C_j(x))_{j=1}^m$ and $\ell_i := (L_{ij}(\ell_{i-1}))_{j=1}^m$. In Lemma 13, we prove that with probability $\geq 1 - 1/m^{1/4}$, $\overline{\ell_d} = 1$. Since every query Q'_j checks the consistency of a set of gates and if the bit $\ell_{dj} = 1$ we get that with probability $1 - 1/m^{1/4} = 1 - 2^{-\Omega(r)}$, completeness equals 1.

Soundness. We will call a circuit $\Gamma_m(\mathcal{C})$ "good" if the following property holds:

For all layers i, $\forall \ell_i \in \{0,1\}^{w_i}$ such that $\overline{\ell_i} \leq 7/10$, the circuit is such that $\overline{L_{i+1}(\ell_i)} \leq 6/10$. (Recall that $L_{i+1}(z)$ denotes the output of layer L_{i+1} when evaluated on the string z.)

Lemma 13 gives us that,

$$\Pr[\forall \ell_i \text{ with } \overline{\ell_i} \leq 7/10, \overline{L_{i+1}(\ell_i)} \leq 6/10] \geq 1 - 2^{-m/2^i}$$

Taking a union bound over the layers of the circuit, we get that,

$$\begin{aligned}\Pr[\Gamma_m(\mathcal{C}) \text{ is good }] &= \Pr[\forall i, \forall \ell_i \text{ with } \overline{\ell_i} \leq 7/10, \overline{L_{i+1}(\ell_i)} \leq 6/10] \\ &\geq 1 - (\log \log m) 2^{-m/2^d} \\ &\geq 1 - 2^{-\sqrt{m}} = 1 - 2^{-\Omega(r)}\end{aligned}$$

We will now show that if the randomized circuit $\Gamma_m(\mathcal{C})$ is good then the new PCP is sound. Since the circuit is good with high probability, showing this is enough to complete the randomized reduction claimed in Theorem 2.

From now on, we will assume that the circuit is good. If the original proof system $\mathcal{P}$ had soundness 6/10, then for all proofs Π that the prover might give, Π satisfies $\leq 6/10$ of the clauses $\mathcal{C}$. Let $\Pi' = (x, \ell_0, \ldots, \ell_d)$ be the proof provided by the new prover.

Let $z_0 := (C_j(x))_{j=1}^m$ and $z_{i+1} := (L_{(i+1)j}(\ell_i))_{i=1}^{w_i}$ be the true local evaluations. Note here that, z_{i+1} is the evaluation bits of layer L_{i+1} evaluated on the bits that the prover provides in the previous layer, ℓ_i. By the soundness of $\mathcal{P}$ we get that x satisfies at most 6/10 of $\mathcal{C}$ which means that $\overline{z_0} \leq 6/10$.

Now we have two cases:

1. The prover provided the bit vectors ℓ_i such that they agree with the true evaluations z_i in most places, i.e.

$$\forall i, \Pr_{j\sim[w_i]}[\ell_{ij} \neq z_{ij}] \leq 1/10.$$

   Hence we have that $\overline{\ell_0} \leq \overline{z_0} + 1/10 \leq 7/10$. Lemma 14 gives us that for $\overline{\ell_i} \leq 7/10$, $\overline{L_{i+1}(\ell_i)} \leq 6/10$ and therefore $\overline{z_{i+1}} \leq 6/10$. Hence we get that by induction, for all i, $\overline{z_i} \leq 6/10$ and $\overline{\ell_i} \leq 7/10$, and more importantly $\overline{\ell_d} \leq 7/10$. Recall that our verifier checks are uniform over the every layer, and since $\ell_{dj} = 1$ is required for verifier's j^{th} check, Q_j to succeed, we get that soundness is $\leq 7/10$.
2. There exists a layer $i \in \{0, \ldots, d\}$ such that:

$$\Pr_{j\sim[w_i]}[\ell_{ij} \neq z_{ij}] > 1/10.$$

   Since z_{ij}'s are the correct evaluations, the above implies that, the prover's proof will fail the local checks in 1/10-fraction of the gates of layer i. Since the verifier checks are uniform over the gates of every layer, (i.e. they check the gate of each layer with the same probability), the verifier checks the incorrect gates with probability at least 1/10. Hence the soundness in this case is $\leq 9/10$.

Note that one of these cases has to occur, hence the overall soundness is the maximum of the two cases, i.e. $\leq 9/10$.

Proof Length. Every layer L_i has width $m/2^i$. Thus the total number of gates in the circuit is $m + m/2 + \ldots = O(m) = O(2^r)$. Since Π' consists of the original proof appended with the circuit evaluations, the proof length is $n + O(2^r)$. ◀

We now complete the proofs of completeness and soundness claims used in the proof of Theorem 2.

▶ **Lemma 13** (Completeness). *Let $y_0 \in \{0,1\}^m$ be such that $\overline{y_0} \geq 9/10$. Let $y_i \in \{0,1\}^{w_i}$ denote the output string of layer i, when $\mathcal{C}$ is evaluated on y_0. Then we have that with probability $\geq 1 - 1/m^{1/4}$ for all i, y_i satisfies $\overline{y_i} \geq 1 - \left(\frac{1}{10}\right)^{2^i}$ and hence $\overline{y_d} = 1$.*

Notice here that the completeness $1-\eta$ increases to $1-(\eta)^2$ at each step, instead of $1-\eta$ to $1-\eta/2$, like it did in the previous section. This increase allows us to use only $\log\log m$ layers to get perfect completeness, albeit with high probability. Now we prove the lemma.

Proof. The theorem statement is implied by proving that with probability $\geq 1 - 1/m^{1/4}$ for all i, $(1 - \overline{y_{i+1}}) \leq (1 - \overline{y_i})^2$.

We will prove the lemma by induction on i. Note that the base case $i = 0$, holds trivially. Now consider the $(i+1)^{th}$ layer of the circuit and the gates $L_{(i+1)j}$ that take as input the set $S_{(i+1)j}$ corresponding to random sets of size f from $[w_i]$.

By induction $\overline{y_i} \geq 1 - \left(\frac{1}{10}\right)^{2^i} \geq .9$ and $.2/(1-\overline{y_i}) \geq 2$. For a fixed $L_{(i+1)j}$, by the Chernoff bound 2 on number of 0's we get,

$$\Pr[L_{(i+1)j}(y_i) = \mathrm{Thr}_{.8}(y_i|_{S_{(i+1)j}}) = 0] = \Pr[\mathrm{Thr}_{.2}((1-y_i)|_{S_{(i+1)j}}) = 1]$$

$$\leq exp(\Omega((.2/(1-\overline{y_i}))\log((.2/(1-\overline{y_i}))(1-\overline{y_i})f))$$
$$= exp(\Omega(\log((.2/(1-\overline{y_i}))f))$$
$$\leq (1-\overline{y_i})^3$$

for some large enough constant f.

Chernoff bound 1 over all the gates in L_{i+1} for the number of 0's gives gives that,

$$\Pr[(1-\overline{y_{i+1}}) \geq (1-\overline{y_i})^2] < exp(-\Omega(((1-\overline{y_i})^2/(1-\overline{y_i})^3)(1-\overline{y_i})^3(m/2^i))) = exp(-\Omega((1-\overline{y_i})^2(m/2^i)))$$

As we have $\log\log m$ layers $m/2^i > m/\log m$, hence

$$\Pr[(1-\overline{y_{i+1}}) \geq (1-\overline{y_i})^2] < exp(-\Omega((1-\overline{y_i})^2(m/\log m)))$$

A Markov bound over all the gates in L_{i+1} for the number of 0's gives gives that,

$$\Pr[(1-\overline{y_{i+1}}) \geq (1-\overline{y_i})^2] \leq (1-\overline{y_i}).$$

Together these bounds imply

$$\Pr[(1-\overline{y_{i+1}}) \geq (1-\overline{y_i})^2] \leq \log^2(m)/\sqrt{m}.$$

Union bound over all $\log m$ layers gives probability $\leq (\log\log m)\log^2(m)/\sqrt{m} \leq 1/m^{1/4}$. Hence with probability $\geq 1-1/m^{1/4}$, $\overline{y_d} \geq 1-\left(\frac{1}{10}\right)^{2^{\log\log(m)}} \geq 1-1/m^2$. As there are $\leq m$ gates at last layer this means with probability $\geq 1-1/m^{1/4}$, $\overline{y_d} = 1$. ◀

▶ **Lemma 14** (Soundness). *Let $y_i \in \{0,1\}^{w_i}$ denote an instantiation of the output gates of layer i with $\overline{y_i} \leq 7/10$. Let $L_{i+1}(y_i)$ denote the output of layer $i+1$ when evaluated on the string y_i. Then with probability $1-2^{-m/2^i}$, for all y_i, $L_{i+1}(y_i)$ satisfies $\overline{L_{i+1}(y_i)} \leq 6/10$. Formally,*

$$\Pr[\forall y_i \text{ with } \overline{y_i} \leq 7/10, \overline{L_{i+1}(y_i)} \leq 6/10] \geq 1-2^{-m/2^i}.$$

Proof. Fix a gate $L_{(i+1)j}$. Given that the fraction of 1s in layer i is at most $7/10$, using Chernoff bound 1, we get that,

$$\Pr[\mathrm{Thr}_{0.8}(S_{(i+1)j}) = 1] = \Pr[\frac{1}{f}\sum_{k\in S_{(i+1)j}} \ell_{ik} - 7/10 > 8/10 = 7/10(1+1/7)] < exp(-(1/7)^2(7f/10)/3) < 1/f$$

for large enough constant f.

By applying Chernoff bound 2 (assuming large enough f) over all gates $L_{(i+1)j}$, we get that,

$$\Pr[\overline{L_{i+1}(y_i)} > 6/10] < exp(-\Omega((6f/10)(\log(6f/10))(m/(f2^{i+1}))))$$
$$= exp(-\Omega((\log(6f/10))(m/2^i)))$$
$$< exp(-2m/2^i).$$

for large enough constant f.

Hence a union bound over all possible $2^{m/2^i}$ strings y_i gives that,

$$\Pr[\exists y_i, \overline{L_{i+1}(y_i)} > 6/10] \leq 2^{m/2^i}e^{-2m/2^i} < 2^{-m/2^i}.$$ ◀

5 Gap-ETH without perfect completeness

In this section we study the relation between the time complexities of the MAX 3-SAT problem with and without perfect completeness. We show that the Gap-ETH conjecture with and without perfect completeness is equivalent by giving an algorithm for MAX 3-SAT without perfect completeness, that uses an algorithm for MAX 3-SAT with perfect completeness as a subroutine and runs in $2^{o(n)}$-time iff the latter does so.

We first prove that Gap-ETH conjecture with and without perfect completeness are equivalent for randomized algorithms with two-sided error. We show this by showing that the Gap-ETH conjecture without perfect completeness is false if the one with perfect completeness is false.

▶ **Theorem 15.** *If there exists a randomized (two-sided error) $2^{o(n)}$ time algorithm for MAX 3SAT$(1, 1-\gamma)$, for all constants $\gamma > 0$, then there exists a randomized (two-sided error) $2^{o(n)}$ time algorithm for MAX 3SAT$(s(1+\epsilon), s)$ for all constants s, ϵ.*

We will prove the above in its contra-positive form. Suppose there is a $2^{o(n)}$ algorithm for MAX 3SAT$(1, 1-\gamma)$ for all constants γ. We will then show that for all constants ϵ, s, δ, there exists an algorithm for MAX 3SAT$(s(1+\epsilon), s)$ with running time less than $2^{\delta n}$. Our randomized algorithm for MAX 3SAT$(s(1+\epsilon), s)$ will use the algorithm for satisfiable MAX 3SAT$(1, 1-\gamma)$ as a subroutine and run in time less than $2^{\delta n}$. The following lemma forms the crux of the proof.

▶ **Lemma 16.** *For all constant $s, \epsilon > 0$ there exists a large enough constant k, such that there exists a randomized reduction from MAX 3-SAT$(s(1+\epsilon), s)$ on n variables and $O(n)$ clause to MAX $O(k)$-CSP$(1, 1/2)$ on n variables and $O(n)$ variables, such that:*

- *If the original instance was a NO instance, then the reduction produces an instance which is not a NO instance with probability $\leq 2^{-n}$.*
- *If the original instance was a YES instance, then the reduction produces a YES instance with probability $\geq 2^{-n/k}$.*

Proof. Let $\mathcal{C} = \{C_1, \ldots, C_m\}$ be a MAX 3SAT$(s(1+\epsilon), s)$ instance. We can assume without loss of generality, that $\epsilon < 1/100$, since the result for a smaller gap implies the result for a larger gap.

Let $(S_i)_{i=1}^{n}$ be a set family in which every set S_i is a random set chosen by sampling with replacement from $[m]$. Consider new clauses B_i such that each clause is a threshold gate: $B_i = \mathrm{Thr}_{s(1+\epsilon/2)}(\mathcal{C}|_{S_i})$, where $\mathcal{C}$ denotes the vector $(C_1(x), \ldots, C_m(x))$.

Our final CSP will be over the original set of variables x_i. We will have a clause for each of the n B_i's. For the i^{th} clause B_i, we will find the values of all C_j such that $j \in S_i$ and then verify that their threshold value is $\geq s(1+\epsilon/2)$. Our query size is $3k$ as we find values for variables in k clauses each of them on 3 variables.

Soundness. For a NO instance and a fixed assignment x the fraction of clauses satisfied by x is $\leq s$. By the Chernoff bound 1, the probability that clause B_i is satisfied is $\leq exp(-(\varepsilon/2)^2 sk/3)$. The probability that at least half of the B_i's are satisfied is at most, $\binom{n}{n/2} exp(-\Omega(\varepsilon^2 skn))$ which is less than $exp(-2n)$, when k is taken to be a large enough constant, depending only on ϵ, s. Therefore by a union bound, the probability that there exists an assignment x that satisfies at least half of the B_i's is $\leq 2^n exp(-2n) \leq 2^{-n}$.

Completeness. For a YES instance there exists an assignment x that satisfies $\geq s(1+\epsilon)$-fraction of the clauses. By the Chernoff bound 1 the probability that the clause B_i is

unsatisfied is $\leq exp(-(\varepsilon/3)^2 sk/2)$ as $\epsilon < 1/100$. Therefore the probability that all the B_i's are satisfied is $(1 - exp(-\Omega(\varepsilon^2 sk)))^n \geq (1 - 10/k)^n$ which is $\geq 2^{-n/k}$ when k is a large enough constant. ◀

Proof of Theorem 15. The randomized algorithm for solving MAX 3SAT$(s(1+\epsilon), s)$ is as follows: We will run the reduction from Lemma 16 $2^{n/k}n^2$ times and then convert the resulting MAX $O(k)$-CSP$(1, 1/2)$ instances to MAX 3SAT$(1, 1-\gamma)$ instances on $O(k2^k n)$ variables and $O(k2^k n)$ clauses where γ is a constant depending on k. Then we run the $2^{o(n)}$ algorithm for MAX 3SAT$(1, 1-\gamma)$ (still $2^{o(n)}$ as k, γ are constants) on the resulting instances and if any of the outputs is YES we will also output YES.

By repeating the algorithm for MAX 3SAT$(1, 1-\gamma)$ poly(n) times we can assume the the probability that the algorithm errs is $\leq 2^{-n^2}$, hence we will assume this wlog.

$$\begin{aligned}\Pr[\text{Error on a YES instance}] &\leq \Pr[\text{Algorithm errs on one of the produced instances}] \\ &\quad + \Pr[\text{None of the } 2^{n/k}n^2 \text{ runs produce a YES instance}] \\ &\leq 2^{-n^2}2^{n/k}n^2 + (1 - 2^{-n/k})^{2^{n/k}n^2} \\ &\leq 2^{-n/2}\end{aligned}$$

$$\begin{aligned}\Pr[\text{Error on a NO instance}] &\leq \Pr[\text{Algorithm errs on one of the produced instances}] \\ &\quad + \Pr[\text{On one of the } 2^{n/k}n^2 \text{ runs the output was not a NO instance}] \\ &\leq 2^{-n^2}2^{n/k}n^2 + 2^{n/k}n^2 2^{-n} \\ &\leq 2^{-n/2}\end{aligned}$$

Total running time $= 2^{n/k}n^2 2^{o(n)}$ which for large enough k is $< 2^{\delta n}$. This gives us the desired contradiction. ◀

We now prove that in fact Gap-ETH conjecture with and without perfect completeness are equivalent for randomized algorithms with no false positives.

▶ **Reminder of Theorem 3.** *If there exists a randomized (with no false positives) $2^{o(n)}$ time algorithm for MAX 3SAT$(1, 1-\gamma)$ for all constant $\gamma > 0$ then there exists a randomized(with no false positives) $2^{o(n)}$ time algorithm for MAX 3SAT$(s(1+\epsilon), s)$ for all constants $s, \epsilon > 0$.*

As in the proof of Theorem 15, we will prove by the above by contradiction. Assume there exists no algorithm for MAX 3SAT$(s(1+\epsilon), s)$ with running time less than $2^{\delta n}$ for some constant $\delta > 0$, while there is a $2^{o(n)}$ algorithm for MAX 3SAT$(1, 1-\gamma)$ for all constants γ. We then give a randomized algorithm for MAX 3SAT$(s(1+\epsilon), s)$ using the algorithm for satisfiable MAX 3SAT$(1, 1-\gamma)$ as a subroutine running in time less than $2^{\delta n}$. The following lemma which is a stronger version of Lemma 16 with only one-sided error forms the crux of the proof.

▶ **Lemma 17.** *For all constant $s, \epsilon > 0$ there exists a large enough constant k, such that there exists a randomized reduction from MAX 3SAT$(s(1+\epsilon), s)$ to MAX $O(k)$-CSP$(1, 1/2)$ with $O(n)$ variables such that:*

- *If the original instance was NO then the reduced instance is also a NO instance.*
- *If the original instance was YES then the reduced instance is YES with probability $\geq 2^{-n/k}$.*

Proof of Theorem 3. The randomized algorithm for solving MAX 3SAT$(s(1+\epsilon), s)$ is as follows: We will run the reduction from Lemma 17 $2^{n/k}n^2$ times and then convert the resulting MAX $O(k)$-CSP$(1, 1/2)$ instances to a MAX 3SAT$(1, 1-\gamma)$ instances on $O(k2^k n)$

variables and $O(k2^k n)$ clauses where γ is a constant depending on k. Then we run the $2^{o(n)}$ algorithm for MAX 3SAT$(1, 1-\gamma)$ algorithm (still $2^{o(n)}$ as k, γ are constants) on them and if any of the outputs is YES we will also output YES.

By repeating the algorithm poly(n) times we can assume the the probability that the algorithm errs (one sided error) is $\leq 2^{-n^2}$, hence we will assume this wlog.

For a NO original instance we will always output a NO instance.

$$\begin{aligned}\Pr[\text{Error on a YES instance}] &\leq \Pr[\text{Algorithm errs on one of the produced instances}] \\ &\quad + \Pr[\text{None of the } 2^{n/k}n^2 \text{ runs produce a YES instance}] \\ &\leq 2^{-n^2}2^{n/k}n^2 + (1-2^{-n/k})^{2^{n/k}n^2)} \\ &\leq 2^{-n}\end{aligned}$$

Total running time $= 2^{n/k}n^2 2^{o(n)}$ which for large enough k is $< 2^{\delta n}$. This gives us the desired contradiction. ◀

Proof of Lemma 17. Let $\mathcal{C} = \{C_1, \ldots, C_m\}$ be a MAX 3SAT$(s(1+\epsilon), s)$. We can assume without loss of generality, that $\epsilon < 1/100$, since the result for a smaller gap implies the result for a larger gap. Let the number of clauses in $\mathcal{C}$ be $m = \rho n$. We will sample with repetition from $\mathcal{C}$ to produce a list L of clauses of size $t\rho n$, for some $t > 1$. We call a list *balanced* if:

1. For every set $S \subseteq \mathcal{C}, |S| = s\rho n$, each clause in S occurs in L at most $s(1+\epsilon/3)t\rho n$ times.
2. For every set $S \subseteq \mathcal{C}, |S| = s(1+\epsilon)\rho n$, each clause in S occurs in L at least $s(1+2\epsilon/3)t\rho n$ times.

It is easy to see that the probability of sampling an unbalanced list is:

$$\begin{aligned}\Pr[\text{L is unbalanced}] &\leq \binom{\rho n}{s\rho n} exp(-\epsilon^2 st\rho n/9) + \binom{\rho n}{s(1+\epsilon)\rho n} exp(-\epsilon^2 s(1+\epsilon)t\rho n/16) \\ &\leq exp(-10\rho n),\end{aligned}$$

when t is large enough and since $\epsilon < 1/100$.

Let $\mathcal{C}'$ be the CSP given by the set of clauses in L (repeated clauses might be present in $\mathcal{C}$). If L is balanced then the soundness of $\mathcal{C}'$ is $\leq s(1+\epsilon/3)$ and completeness is $\geq s(1+2\epsilon/3)$. If our list is not balanced we will reject it and output any NO instance. This can be done in polynomial time as we can check the condition 1 by finding a set of clauses of size $s\rho n$ which occurs the most and checking that it occurs at most $s(1+\epsilon/3)t\rho n$ in L. We can similarly check condition 2.

Let $(S_i)_{i=1}^{|L|}$ be the set family given to us by the expander sampler from Lemma 7 with parameters $\mathcal{ES}((100/(s^2\epsilon^2 k)), (s\epsilon/6), |L|)$. Consider new clauses B_i such that each clause is a threshold gate, i.e. $B_i = \text{Thr}_{s(1+\epsilon/2)}(\mathcal{C}'|_{S_i})$, where $\mathcal{C}'$ denotes the vector of clauses of L. By the sampler property $|S_i| \leq k$ and the number of B_i's is equal to $|L| = t\rho n$.

Our final CSP will be given by the set of clauses B_i. For the i^{th} clause will find the values of all C'_j such that $j \in S_i$ and then verify that their threshold value is $\geq s(1+\epsilon/2)$. Our query size is $3k$ as we find values for variables in k clauses each of them on 3 variables.

Soundness . If L is balanced, in the NO case the soundness is $\leq s(1+\epsilon/3)$. Then we get that,

$$\Pr_i[B_i(\mathcal{C}') = 1] = \Pr\left[\frac{1}{|S_i|}\sum_{j\in S_i}\mathcal{C}'_j \geq s(1+\epsilon/2)\right]$$
$$= \Pr\left[\frac{1}{|S_i|}\sum_{j\in S_i}\mathcal{C}'_j - s(1+\epsilon/3) \geq s(\epsilon/6)\right]$$
$$\leq 100/(s^2\epsilon^2 k)$$

where the last inequality follows from the properties of expander sampler in Lemma 7. Now for large enough k we get $100/(s^2\epsilon^2 k) \leq 1/2$ hence starting from all NO instances gives us NO instances.

If L is unbalanced we always output a NO instance.

Completeness. By the property of the expander sampler in Lemma 7, the number of query sets that intersect with some query set S_i are at most $O(k^2)$ for large enough k. As the original instance was a YES instance there exists an $x = x_c$ which satisfies $s(1+\epsilon)$ fraction of the clauses. As each clause of list L is a random clause from the original set of clauses, the probability that any specific B_i evaluates to 1 is $\geq 1 - exp(-\Omega(-\epsilon^2 ks))$ by the Chernoff bound for assignment x_c.

As each clause of list L is a random clause from the original set of clauses, we get that the random variables (randomness from choosing the list L, after fixing the sets S_i) B_i and B_j are independent, if two query sets S_i and S_j do not intersect. As calculated above, the probability that any clause fails is $\leq exp(-\Omega(-\epsilon^2 ks))$. For large enough constants k,

$$e \cdot O(k^2) exp(-\Omega(-\epsilon^2 ks)) < 1,$$

which allows us to apply the Lovász local lemma as given in Lemma 5. This gives us that,

$$\Pr_L[\wedge B_i(\mathcal{C}') = 1] \geq (1 - 1/k^3)^{t\rho n} \geq 2^{-n/(2k)}.$$

Taking into account the case where L is unbalanced, the probability of outputting a YES instance is $\geq 2^{-n/(2k)} - 2^{-10\rho n} \geq 2^{-n/k}$ for large enough k. ◀

6 Conclusion

The reduction in Section 3 is not useful to get perfect completeness for PCPs, while preserving their query complexity and losing some factor in the randomness complexity. When the construction is composed with query reduction, it only gives us that $\text{PCP}_{c,s}[\log n + O(1), O(1)] \subseteq \text{PCP}_{1,s'}[\log n + O(\log\log n), O(1)]$, which is anyway the blow-up incurred in state of the art PCPs for NP [11]. Hence we pose the following problem:

▶ **Open Problem 1.** *Let $c, s, s' \in (0,1)$ with $s < c$ be constants. Then is it true that,*

$$PCP_{c,s}[\log n + O(1), O(1)] \subseteq PCP_{1,s'}(\log n + o(\log\log n), O(1))?$$

References

1 Divesh Aggarwal and Noah Stephens-Davidowitz. (Gap/S)ETH hardness of SVP. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 228–238, 2018.

2 Benny Applebaum. Exponentially-Hard Gap-CSP and Local PRG via Local Hardcore Functions. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 836–847, 2017.

3 Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof Verification and the Hardness of Approximation Problems. *J. ACM*, 45(3):501–555, 1998. doi:10.1145/278298.278306.

4 Mohammad Bavarian, Thomas Vidick, and Henry Yuen. Hardness amplification for entangled games via anchoring. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 303–316, 2017.

5 Mihir Bellare, Oded Goldreich, and Madhu Sudan. Free Bits, PCPs, and Nonapproximability-Towards Tight Results. *SIAM J. Comput.*, 27(3):804–915, 1998. doi:10.1137/S0097539796302531.

6 Eli Ben-Sasson, Yohay Kaplan, Swastik Kopparty, Or Meir, and Henning Stichtenoth. Constant Rate PCPs for Circuit-SAT with Sublinear Query Complexity. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 320–329, 2013.

7 Eli Ben-Sasson and Madhu Sudan. Short PCPs with Polylog Query Complexity. *SIAM J. Comput.*, 38(2):551–607, 2008. doi:10.1137/050646445.

8 Arnab Bhattacharyya, Suprovat Ghoshal, Karthik C. S., and Pasin Manurangsi. Parameterized Intractability of Even Set and Shortest Vector Problem from Gap-ETH. In *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, pages 17:1–17:15, 2018.

9 Parinya Chalermsook, Marek Cygan, Guy Kortsarz, Bundit Laekhanukit, Pasin Manurangsi, Danupon Nanongkai, and Luca Trevisan. From Gap-ETH to FPT-Inapproximability: Clique, Dominating Set, and More. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 743–754, 2017.

10 Siu On Chan. Approximation resistance from pairwise independent subgroups. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 447–456, 2013.

11 Irit Dinur. The PCP theorem by gap amplification. *J. ACM*, 54(3):12, 2007. doi:10.1145/1236457.1236459.

12 Irit Dinur. Mildly exponential reduction from gap 3SAT to polynomial-gap label-cover. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:128, 2016. URL: http://eccc.hpi-web.de/report/2016/128.

13 Irit Dinur, Subhash Khot, Guy Kindler, Dor Minzer, and Muli Safra. Towards a proof of the 2-to-1 games conjecture? In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 376–389, 2018.

14 Irit Dinur and Pasin Manurangsi. ETH-Hardness of Approximating 2-CSPs and Directed Steiner Network. In *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, pages 36:1–36:20, 2018.

15 David Gillman. A Chernoff Bound for Random Walks on Expander Graphs. *SIAM J. Comput.*, 27(4):1203–1220, 1998. doi:10.1137/S0097539794268765.

16 Oded Goldreich. A Sample of Samplers - A Computational Perspective on Sampling (survey). *Electronic Colloquium on Computational Complexity (ECCC)*, 4(20), 1997. URL: http://eccc.hpi-web.de/eccc-reports/1997/TR97-020/index.html.

17 Venkatesan Guruswami and Luca Trevisan. The Complexity of Making Unique Choices: Approximating 1-in- k SAT. In *Approximation, Randomization and Combinatorial Optimization, Algorithms and Techniques, 8th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, APPROX 2005 and 9th InternationalWorkshop on Randomization and Computation, RANDOM 2005, Berkeley, CA, USA, August 22-24, 2005, Proceedings*, pages 99–110, 2005.

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

19 Russell Impagliazzo and Ramamohan Paturi. On the Complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001. doi:10.1006/jcss.2000.1727.

20 Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings on 34th Annual ACM Symposium on Theory of Computing, May 19-21, 2002, Montréal, Québec, Canada*, pages 767–775, 2002.

21 Pasin Manurangsi and Prasad Raghavendra. A Birthday Repetition Theorem and Complexity of Approximating Dense CSPs. In *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, pages 78:1–78:15, 2017.

22 Anup Rao. Parallel Repetition in Projection Games and a Concentration Bound. *SIAM J. Comput.*, 40(6):1871–1891, 2011. doi:10.1137/080734042.

23 Ran Raz. A Parallel Repetition Theorem. *SIAM J. Comput.*, 27(3):763–803, 1998. doi:10.1137/S0097539795280895.

A Proof of Theorem 8

We first consider an expander sampler from Lemma 7 with parameters $(\epsilon, \delta/2, N)$. This is an expander graph G over N nodes, with the set family $\mathcal{ES} = (S_v)_{v\in[N]}$, where S_v = set of neighbors of v in G. From the proof given in the sampler survey [16], one can check that if one takes the second eigenvalue λ to be small enough, $\leq \text{poly}(\epsilon, \delta)$, then the following holds:

For any string $x \in \{0,1\}^N$,

$$\Pr_{S\sim\mathcal{ES}}[|\overline{(x|_S)} - \overline{x}| > \epsilon] \leq \delta/2. \tag{1}$$

Now let us see, how to achieve property (2). We use the Expander Mixing Lemma, and show (proof deferred to later in this section) that if λ is small enough ($\leq \text{poly}(\gamma)$) then the following holds: For all $\eta < (1-\gamma)/2$, for any string $x \in \{0,1\}^N$, where $\overline{x} \geq 1-\eta$, we get that,

$$\Pr_{S\sim\mathcal{ES}}[\overline{(x|_S)} < \gamma] \leq \eta/4. \tag{2}$$

Taking the second eigenvalue λ less than the minimum required in both proofs above, we get that both the above statements hold, for some $\lambda = O_{\epsilon,\delta,\gamma}(1)$. Note that the degree of an expander, which is also the sample complexity, is $\text{poly}(1/\lambda) = O_{\epsilon,\delta,\gamma}(1) = C$, hence property (3) holds.

To get property (4), we arbitrarily take $N/2$ of the samples and define this as the set family $\mathcal{S}$ given by the sampler. This hurts the probabilities in equations 1 and 2 by a factor of at most 2 and hence we get properties (1), (2) for the new set family.

Proof of property (2). Let $B \subset [N]$ be the positions of zeros in x and let $C \subset [N]$ be the set of vertices that have at least $1-\gamma$-fraction of their neighbors in B. Notice that the vertices S in C are exactly those samples on which $\overline{x|_S} < 1-\gamma$, hence it is enough to bound $|C|/N = \eta'$.

We know that $|B|/N = \eta$ and let $|C|/N = \eta'$. By the Expander mixing lemma we get that,

$$\frac{|E(B,C)|}{|E(G)|} \leq \eta\eta' + \lambda\sqrt{\eta\eta'},$$

where λ is the second eigenvalue of graph G. But by the property of C, we get, $\frac{|E(B,C)|}{|E(G)|} \geq (1-\gamma)\eta'$. Combining these two we get that $\eta' \leq (\lambda^2/(1-\gamma-\eta)^2)\eta \leq (4\lambda^2/(1-\gamma)^2)\eta$. We can take λ to be small enough in terms of γ, to get that $\eta' < \eta/4$. ◀