32nd Computational Complexity Conference

CCC 2017, July 6–9, 2017, Riga, Latvia

Edited by

Ryan O'Donnell

Editor

Ryan O'Donnell
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA
USA
odonnell@cs.cmu.edu

ACM Classification 1998
F.1 Computation by Abstract Device

ISBN 978-3-95977-040-8

Published online and open access by
Schloss Dagstuhl – Leibniz-Zentrum für Informatik GmbH, Dagstuhl Publishing, Saarbrücken/Wadern, Germany. Online available at http://www.dagstuhl.de/dagpub/978-3-95977-040-8.

Publication date
July, 2017

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

Digital Object Identifier: 10.4230/LIPIcs.CCC.2017.0

ISBN 978-3-95977-040-8 **ISSN 1868-8969** **http://www.dagstuhl.de/lipics**

LIPIcs – Leibniz International Proceedings in Informatics

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

ISSN 1868-8969

http://www.dagstuhl.de/lipics

Contents

32nd Computational Complexity Conference (CCC 2017).
Editor: Ryan O'Donnell
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
COMPUTATIONAL COMPLEXITY CONFERENCE

Preface

The papers in this volume were accepted for presentation at the 32nd Computational Complexity Conference (CCC 2017), held July 6 to July 9 in Riga, Latvia. The conference is organized by the Computational Complexity Foundation (CCF) in cooperation with the European Association for Theoretical Computer Science (EATCS) and the ACM Special Interest Group on Algorithms and Computation Theory (SIGACT). CCC 2017 is sponsored by Microsoft Research and the University of Latvia.

The call for papers sought original research papers in all areas of computational complexity theory. Of the 98 submissions, the program committee selected 33 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 for their scientific contributions; the board of trustees of the Computational Complexity Foundation, most especially its president Dieter van Melkebeek for extensive advice and assistance; the Local Arrangements Committee chair Andris Ambainis for help with scheduling; Avi Wigderson for contributing three two-hour tutorials on the topic of "Operator Scaling: theory, applications and connections"; and, Marc Herbstritt for coordinating the production of these proceedings.

Ryan O'Donnell
Program Committee Chair, on behalf of the Program Committee

Awards

The program committee of the 32nd Computational Complexity Conference is very pleased to present the **Best Paper Award** to Xi Chen, Rocco A. Servedio, Li-Yang Tan, Erik Waingarten, and Jinyu Xie for their paper

Settling the query complexity of non-adaptive junta testing;

and, the **Best Student Paper Award** to Mrinal Kumar for his paper

A quadratic lower bound for homogeneous algebraic branching programs.

32nd Computational Complexity Conference (CCC 2017).
Editor: Ryan O'Donnell
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
COMPUTATIONAL COMPLEXITY CONFERENCE

Conference Organization

Program Committee

Manindra Agrawal, IIT Kanpur
Eli Ben-Sasson, Technion
Amit Chakrabarti, Dartmouth College
Kai-Min Chung, Academia Sinica
Dmitry Gavinsky, Czech Academy of Sciences
Joshua Grochow, University of Colorado, Boulder and the Santa Fe Institute
Michal Koucký, Charles University
Shachar Lovett, UC San Diego
Ryan O'Donnell (chair), Carnegie Mellon University
Dana Ron, Tel Aviv University
Benjamin Rossman, University of Toronto

Local Arrangements Committee

Andris Ambainis (chair), University of Latvia
Dace Mileika, University of Latvia
Alexander Rivosh, University of Latvia
Raqueline Santos, University of Latvia
Juris Smotrovs, University of Latvia

Board of Trustees

Eric Allender (Treasurer), Rutgers University
Boaz Barak, Harvard University
Sevag Gharibian (Secretary), Virginia Commonwealth University
Venkatesan Guruswami, Carnegie Mellon University
Dieter van Melkebeek (President), University of Wisconsin-Madison
Ryan O'Donnell, Carnegie Mellon University
Rocco A. Servedio, Columbia University
Jacobo Toran, University of Ulm
Osamu Watanabe, Tokyo Institute of Technology

32nd Computational Complexity Conference (CCC 2017).
Editor: Ryan O'Donnell
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
COMPUTATIONAL COMPLEXITY CONFERENCE

External Reviewers

Abuzer Yakaryılmaz
A. C. Cem Say
Adam Bouland
Adam Klivans
Aleksandrs Belovs
Alexander Razborov
Alexander Shen
Amir Shpilka
Andrej Bogdanov
Andris Ambainis
Ankit Garg
Ankit Gupta
Arnab Bhattacharyya
Arvind
Ashley Montanaro
Aviad Rubinstein
Avishay Tal
Avraham Ben-Aroya
Badih Ghazi
Ben Lee Volk
Benny Applebaum
Bill Fefferman
Boaz Barak
Brendan Juba
Bruno Bauwens
Bruno Loff
Cedric Yen-Yu Lin
Chandan Saha
Chi-Ning Chou
Clément Canonne
Cris Moore
Damien Stehle
Dana Moshkovitz
Daniel Dadush
Daniel Kane
David Ellis
David Mix Barrington
David Steurer
David Zuckerman
Debajyoti Bera
Denis Pankratov
Dhiraj Holden
Dieter van Melkebeek
Diptarka Chakraborty
Divesh Aggarwal
Dominik Scheder
Ehud Friedgut
Eiji Miyano
Emanuele Viola
Eric Allender
Eric Blais
Eshan Chattopadhyay
Florian Speelman
François Le Gall
Friedrich Eisenbrand
Gabor Ivanyos
Gil Cohen
Govind Ramnarayan
Greg Valiant
Guoming Wang
Guy Rothblum
Hamed Hatami
Han-Hsuan Lin
Harry Buhrman
Hartmut Klauck
Henry Yuen
Igor Oliveira
Igor Shinkar
Ilan Komargodski
Ilias Diakonikolas
Ilya Volkovich
Inbal Livni Navon
Irit Dinur
Ishay Haviv
Jaiganesh Balasundaram
James Lee
Jeffrey Shallit
Jingbo Liu
Johan Hastad
John M. Hitchcock
Joshua Brody
Justin Thaler
Jyun-Jie Liao
Karl Wimmer
Kevin Woods
Lance Fortnow
Li-Yang Tan
Madhu Sudan
Makrand Sinha
Marius Zimand
Mark Braverman
Massimo Lauria
Michael A. Forbes
Michael Bremner
Michael Saks
Mika Göös
Mike Saks
Moran Feldman
Mrinal Kumar
Navid Talebanfard
Neeraj Kayal
Nengkun Yu
Nikolay Vereshchagin
Nisheeth Vishnoi
Nitin Saurabh
Nitin Saxena
Niv Buchbinder
Noga Ron-Zewi
Oded Goldreich
Oded Regev
Omri Weinstein
Or Meir
Parikshit Gopalan
Partha Mukhopadhyay
Pavel Pudlak
Peter Burgisser
Peter Høyer
Pierre McKenzie
Prahladh Harsha
Pravesh Kothari
Rafael Oliveira
Raghu Meka
Rahul Jain
Rahul Santhanam
Rajat Mittal
Ralph Bottesch
Ramprasad Saptharishi
Ran Gelles
Raúl García-Patrón Sánchez
Reut Levi
Richard Peng
Rishi Saket
Robert Robere
Robin Kothari
Rocco Servedio
Roei Tell
Rohit Gurjar
Ronald de Wolf
Ryan Williams
Sagar Kale
Sam Buss
Santosh Vempala
Scott Aaronson
Sevag Gharibian
Shalev Ben-David
Shengyu Zhang
Shubhangi Saraf
Shuichi Hirahara
Siu On Chan
Sivakanth Gopi
Srikanth Srinivasan
Standa Živný
Stephen Fenner
Steven Heilman
Subhash Khot
Suman Kalyan Bera
Susanna Figueiredo De Rezende
Sushant Sachdeva
T. S. Jayram
Talya Eden
Thomas Watson
Troy Lee
Valentine Kabanets
Venkata Gandikota
Venkatesan Guruswami
Venkatesh Medabalimi
Vitaly Feldman
Weiqiang Wen
Wolfgang Mulzer
Xi Chen
Xiaodi Wu
Yin-Hsun Huang
Youming Qiao
Young Kun Ko
Yuval Filmus

32nd Computational Complexity Conference (CCC 2017).
Editor: Ryan O'Donnell
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
COMPUTATIONAL COMPLEXITY CONFERENCE

Random Resolution Refutations*†

Pavel Pudlák[1] and Neil Thapen[2]

1 **Institute of Mathematics, Czech Academy of Sciences, Prague, Czech Republic**
`pudlak@math.cas.cz`
2 **Institute of Mathematics, Czech Academy of Sciences, Prague, Czech Republic**
`thapen@math.cas.cz`

Abstract

We study the *random resolution* refutation system defined in [Buss et al. 2014]. This attempts to capture the notion of a resolution refutation that may make mistakes but is correct most of the time. By proving the equivalence of several different definitions, we show that this concept is robust. On the other hand, if $\mathbf{P} \neq \mathbf{NP}$, then random resolution cannot be polynomially simulated by any proof system in which correctness of proofs is checkable in polynomial time.

We prove several upper and lower bounds on the width and size of random resolution refutations of explicit and random unsatisfiable CNF formulas. Our main result is a separation between polylogarithmic width random resolution and quasipolynomial size resolution, which solves the problem stated in [Buss et al. 2014]. We also prove exponential size lower bounds on random resolution refutations of the pigeonhole principle CNFs, and of a family of CNFs which have polynomial size refutations in constant depth Frege.

1998 ACM Subject Classification F.4.1 Mathematical Logic, F.1.3 Complexity Measures and Classes

Keywords and phrases Proof complexity, random, resolution

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.1

1 Introduction

The following system for refuting propositional CNFs was introduced in [3]. Let F be a CNF in variables $x_1, \ldots, x_n$ and let $0 < \varepsilon < 1$.

▶ **Definition 1.** An *ε-random resolution distribution*, or *ε-RR distribution*, of F is a probability distribution $\mathcal{D}$ on pairs $(B_i, \Pi_i)_{i \sim \mathcal{D}}$ such that

1. for each $i \in \mathcal{D}$, B_i is a CNF in variables $x_1, \ldots, x_n$ and Π_i is a resolution refutation of $F \wedge B_i$
2. for every $\alpha \in \{0,1\}^n$, $\Pr_{i \sim \mathcal{D}}[B_i \text{ is satisfied by } \alpha] \geq 1 - \varepsilon$.

The *size* and the *width* of $\mathcal{D}$ are defined respectively as the maximum size and maximum width of the refutations Π_i (if these maxima exist).

This is sound as a refutational system, in the sense that if F has an ε-RR distribution then F is unsatisfiable. To see this, consider any assignment $\alpha \in \{0,1\}^n$. Since $\varepsilon < 1$, there is at least one pair (B_i, Π_i) such that α satisfies B_i and Π_i is a resolution refutation of $F \wedge B_i$. So α cannot also satisfy F, by the soundness of resolution. The system is also complete,

* The full version of the paper is available as [13], `https://eccc.weizmann.ac.il/report/2016/175/`.

† Partially supported by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC grant agreement 339691. The Institute of Mathematics of the Czech Academy of Sciences is supported by RVO:67985840.

since resolution is complete and we can take $\mathcal{D}$ to consist of a single pair (B, Π) where B is any tautology and Π is a (possibly exponential sized) resolution refutation of F.

On the other hand, as defined, it is not a propositional proof system in the sense of Cook and Reckhow [6], because it is defined by a semantic condition that presumably cannot be tested in polynomial time (see Proposition 10). Nevertheless it makes perfect sense to compare the complexity of proofs in it with proofs in the standard proof systems, in particular with resolution and bounded depth Frege. We prove some results in this direction in this work. Note also that the definition is particular to resolution, and we must take care if we try to generalize it. For example, if we instead define a random Frege distribution system, in which B and Π can contain arbitrary formulas, then we can trivially refute any unsatisfiable F by setting $B = \neg F$.

As with some concepts of probabilistic computation studied in computation complexity theory, one can use the linear programming duality to give an equivalent definition of the system based on probability distributions over inputs rather than over proofs (see Definition 3). This is very useful if one needs to prove lower bounds. Another essentially equivalent formulation is in terms of semantic resolution derivations. This means, roughly speaking, that instead of having an auxiliary formula that is satisfied with high probability, we consider semantic derivations with respect to a large subset of inputs, where lines in the proof are clauses. In a sense, this captures better the intuitive idea of a proof with errors.

Let us also mention that while we tend to think of the error ε as something small, there is a simple amplification lemma that allows us to shrink the error at some cost in proof size. Thus, for the questions we are interested in, without loss of generality we can take $\varepsilon = \frac{1}{2}$.

The definition was first proposed by Stefan Dantchev. Its appearance in [3] is ultimately motivated by an open problem in bounded arithmetic. We will not go into detail about the connection to bounded arithmetic in this abstract, and instead will discuss the problem in terms of constant depth Frege proof systems. One of the longstanding open problems in proof complexity is to prove (or disprove) that the set of polylogarithmic width CNFs with quasipolynomial size refutations in depth i Frege strictly increases as i increases.[1] The simplest instance of this problem is to separate $R(\log)$, which is effectively a low depth Frege system, from higher levels of the constant depth Frege hierarchy. The system $R(\log)$, introduced in [10], is an extension of resolution in which in place of literals one can use small conjunctions, of logarithmic size in the length of the proof. We can separate $R(\log)$ from weaker fragments of constant depth Frege, but not from stronger ones (see for example [14]).

The system $R(\log)$ corresponds to a particular fragment T^2_2 of bounded arithmetic. Since this problem has been notoriously open for many years, it was proposed in [3] to consider, in place of T^2_2, theories of similar strength but of a rather different nature, based on Jeřábek's approximate counting [8]. Interestingly, it turned out that although there are no proof systems associated in the usual sense with these theories, one could use random resolution to prove separation of one of them from higher fragments. It suffices to find a narrow CNF which does not have a narrow 1/2-random resolution distribution, but has a quasipolynomial size refutation in some constant depth Frege system. The problem of separating the theory from other fragments of bounded arithmetic was eventually solved by different means [2], but the problem of proving lower bounds on the width of random resolution distributions remained open.

[1] We emphasize that we are interested in this question for quasipolynomial size proofs. This matches the natural question in bounded arithmetic, and a separation for polynomial size is known [7], using a padded pigeonhole principle $\mathrm{PHP}_{(\log n)^k}$ which has short proofs in some depth i, but is such that the exponential size lower bound for PHP in depth $i-1$ gives a quasipolynomial lower bound for the padded version.

In this paper we solve this problem by proving that the propositional translation of the *coloured polynomial local search* principle CPLS, introduced in [12], which has polynomial size resolution refutations, does not have narrow 1/2-random resolution distributions. Previously, lower bounds on random resolution have only been known for treelike refutations [3] or for relatively small errors ε [11].

The proof is based on a lemma that looks like a rudimentary version of the switching lemmas used in propositional proof complexity (see the discussion at the start of Section 3). Although this does not solve the big open problem of giving a new separation in constant depth Frege, we believe that our result may pave the way to a solution. It has been conjectured that in order to separate constant depth Frege system, we need only to prove switching lemmas for certain more complicated tautologies, generalizations of CPLS (see for example [15]). Nevertheless, all attempts in this direction have failed so far because of the complexity of the associated combinatorial problems. Our proof gives us some hope that eventually it will be possible to prove such lemmas.

The full version of the paper is available as an ECCC technical report [13]. In this extended abstract we present definitions and statements of the main results. We state our theorems without detailed proofs with one exception, which is the simplest lower bound from the full paper and is intended to demonstrate our lower bound technique.

2 Basic properties and alternative definitions

We first introduce some notation. We identify CNF formulas with sets of clauses. We will use 0 (false) and 1 (true) to represent truth values. For a formula F and an assignment α of truth values to its variables, we denote by $F[\alpha]$ the truth value to which the formula is evaluated by α. If ρ is a partial assignment, we denote by F^ρ the formula obtained by substituting ρ into F and simplifying the formula (that is, replacing a conjunction by 0 if one conjunct is 0, etc.).

The *width* of a clause is the number of literals it contains. The *width* and *size* of a refutation are respectively the width of its widest clause and the total number of clauses. A k-CNF is a CNF in which every clause has width at most k.

We will often use the notation $p_1 \wedge \cdots \wedge p_r \to q_1 \vee \cdots \vee q_s$ to stand for the clause $\neg p_1 \vee \cdots \vee \neg p_r \vee q_1 \vee \cdots \vee q_s$, where $p_1, \ldots, p_r, q_1, \ldots, q_r$ can be any literals. In this notation the resolution rule can, for example, have the form: from $A \wedge p \to C$ and $A \wedge \neg p \to D$ conclude $A \to C \vee D$, where p is a literal, A is a conjunction of literals and C and D are clauses.

If p is a literal, we will sometimes write $p = 1$ instead of the literal p and $p = 0$ instead of the literal $\neg p$. Similarly we will write $p \neq 1$ or $p \neq 0$ to mean respectively $\neg p$ or p. If $p_1, \ldots, p_r$ are literals and $\beta \in \{0,1\}^r$ we write $\bar{p} = \beta$ to stand for the conjunction $\bigwedge_{1 \le i \le k} p_i = \beta_i$ where each conjunct is formally either p_i or its negation, as above; and $\bar{p} \neq \beta$ to stand for the disjunction $\bigvee_{1 \le i \le k} p_i \neq \beta_i$.

We write $[n]$ for $\{0, \ldots, n-1\}$. When we formalize combinatorial principles as CNFs, if the principle involves a function $f : [n] \to [m]$ we will often formalize f by introducing variables for its "bit-graph". That is, for each $x < n$ we introduce $\log m$ variables $(f(x))_0, \ldots, (f(x))_{\log m - 1}$ representing the value of $f(x)$ in binary. For the sake of simplicity, in this situation we will assume that m is a power of 2. For $y < m$ we will write $f(x) = y$ to stand for the conjunction $\bigwedge_i (f(x))_i = \beta_i$, where $\beta \in \{0,1\}^{\log m}$ is y written in binary, and we will write $f(x) \neq y$ for the disjunction $\bigvee_i (f(x))_i \neq \beta_i$.

Because we deal with propositional refutation systems, rather than proof systems, for us the natural translation into propositional logic of a true first order principle, such as the

pigeonhole principle PHP, is a family of unsatisfiable CNFs that we want to refute, rather than a family of tautologous DNFs that we want to prove. Therefore we will use the same name, PHP, for both this family of CNFs and the original principle. It should be clear from the context which is meant, and the propositional version will often be written with a size parameter, for example as PHP_n.

In the rest of this section, let F be a CNF in variables $x_1, \ldots, x_n$ and let $0 < \varepsilon < 1$. Our definition of the size of an ε-RR distribution above does not take into account the size of the sample space (that is, of the set of pairs (B, Π) appearing in $\mathcal{D}$) but one can show that the size of the sample space can be bounded, at the cost of slightly increasing the error ε. Also we can decrease the error ε at the cost of increasing the width and size.

▶ **Lemma 2.** *Suppose F has an ε-RR distribution of width w and size s. Then*
1. *it also has 2ε-RR distribution of the same size and width, in which the sample space has size $O(n/\varepsilon)$, and*
2. *for every $k \geq 1$ it also has an ε^k-RR distribution of width at most kw and size $O(s^k)$.*

We will now give two more definitions equivalent to random resolution distributions.

▶ **Definition 3.** Let Δ be a probability distribution on $\{0,1\}^n$. An *(ε, Δ)-random resolution refutation*, or *(ε, Δ)-RR refutation*, of F is a pair (B, Π) such that
1. B is a CNF in variables $x_1, \ldots, x_n$ and Π is a resolution refutation of $F \wedge B$
2. $\Pr_{\alpha \sim \Delta}[B[\alpha] = 1] \geq 1 - \varepsilon$.

This definition is in general not sound, for a fixed Δ. However, if an (ε, Δ)-RR refutation exists for *all* distributions Δ, then this is equivalent to the existence of an ε-RR distribution, as follows.

▶ **Proposition 4.** *The following are equivalent.*
1. *F has an ε-RR distribution of width w and size s.*
2. *F has an (ε, Δ)-RR refutation of width w and size s for every distribution Δ on $\{0,1\}^n$.*

Proof. One direction is just an averaging argument. The other is a consequence of the minimax theorem. ◀

The following generalization of this will be useful for proving lower bounds.

▶ **Proposition 5.** *Proposition 4 still holds if we allow Δ to range over distributions on partial assignments, rather than total assignments. In this case we change item 2 in Definition 3 to* $\Pr_{\rho \sim \Delta}[B[\rho] = 0] \leq \varepsilon$.

Semantic derivations were introduced in [9]. We will use the special case defined by clauses.

▶ **Definition 6.** Let $\mathcal{A} \subseteq \{0,1\}^n$ be a nonempty set of truth assignments. We say that a formula C is a *semantic consequence over $\mathcal{A}$* of formulas $C_1, \ldots, C_r$, written $C_1, \ldots, C_r \vDash^{\mathcal{A}} C$, if every assignment in $\mathcal{A}$ that satisfies $C_1, \ldots, C_r$ also satisfies C.

A *semantic resolution refutation of F over $\mathcal{A}$* is a sequence Π of *clauses*, ending with the empty clause, in which every clause either belongs to F or is a semantic consequence over $\mathcal{A}$ of at most two earlier clauses.

▶ **Definition 7.** Let Δ be a probability distribution on $\{0,1\}^n$. An *(ε, Δ)-semantic refutation* of F is a pair $(\mathcal{A}, \Pi)$ such that
1. Π is a semantic refutation of F over $\mathcal{A}$, and
2. $\Pr_{\alpha \sim \Delta}[\alpha \in \mathcal{A}] \geq 1 - \varepsilon$.

▶ **Proposition 8.** *If F has an (ε, Δ)-RR refutation of width w and size s, then it also has an (ε, Δ)-semantic resolution refutation of width $\leq w$ and size $\leq s$.*

In the opposite direction, if F has an (ε, Δ)-semantic refutation of width w and size s, then it also has an (ε, Δ)-RR refutation of width $O(w)$ and size at most $O(sw^2)$.

We show that random 3-CNFs with sufficiently high density have small RR distributions, while as is well-known, they only have exponentially large resolution refutations [5].

▶ **Proposition 9.** *A random 3-CNF F with n variables and $64n$ clauses has a $1/2$-RR distribution of constant width and constant size with probability exponentially close to 1.*

Proof. With exponentially high probability, F has the property that no single assignment satisfies more than a fraction $15/16$ of its clauses. We define a distribution as follows: choose a clause C_i of the form $y_1 \vee y_2 \vee y_3$ from F uniformly at random, let the auxiliary formula B_i be the CNF $\neg y_1 \wedge \neg y_2 \wedge \neg y_3$, and let Π_i be the three-step derivation of the empty clause from C_i and B_i. Then with high probability this is a $15/16$-RR distribution for F, which can be amplified to a $1/2$-RR distribution using Lemma 2. ◀

This gives a separation of narrow random resolution from resolution in one direction (Theorem 16 will give the opposite direction). We can also prove such a separation for an explicit sequence of CNFs, the *retraction weak pigeonhole principle* (see [4, 8]) that asserts that there is no pair of functions $f : [2n] \to [n]$ and $g : [n] \to [2n]$ such that $g(f(x)) = x$ for all $x < n$.

We can now address the natural question of whether random resolution can be presented as a standard propositional proof system in the sense of Cook and Reckhow [6], or at least whether it can be polynomially simulated by such a system. Because we want to compare other systems with random resolution, we adapt the definition to refutation systems – this makes no difference to the result, since any proof system can be considered as a refutation system and vice versa. The essential property of Cook and Reckhow's definition is that one can test the correctness of refutations in polynomial time, that is, that the binary relation "Π is a refutation of F" is decidable in deterministic polynomial time. The other two properties, soundness and completeness, are satisfied by random resolution.

In order to state our question formally, we must say which object we choose to represent a refutation in random resolution, and what polynomial simulation means. We will consider $1/2$-RR distributions in which all samples have the same weight. Such a distribution can be written down simply as a list of pairs (B_i, Π_i), and by Lemma 2 we do not lose anything important if we only consider $1/2$-RR distributions in this form. Polynomial simulation of refutation systems can be defined in our situation, where correctness may not be decidable in polynomial time, in essentially the same way as for standard refutation systems.

▶ **Proposition 10.** *If $\mathbf{P} \neq \mathbf{NP}$, then $1/2$-RR cannot be polynomially simulated by any Cook-Reckhow refutation system, and in particular is not itself a Cook-Reckhow refutation system.*

Proof. This is a corollary of the PCP Theorem, which can be stated as follows (see Theorem 11.9 in [1]): there exists a polynomial time computable function g and a constant $\delta < 1$ such that for every CNF formula F, $g(F)$ is a 3-CNF formula such that

1. if F is satisfiable, then $g(F)$ is also satisfiable
2. otherwise, every assignment satisfies at most a fraction δ of the clauses of $g(F)$.

Using the construction from the proof of Lemma 9, this implies that if F is unsatisfiable then $g(F)$ has a δ-RR distribution, which furthermore is constructable in polynomial time.

The error δ can be reduced to $1/2$ by Lemma 2 and, again, this can be done in polynomial time. Let h denote the polynomial time computable function h that from a given unsatisfiable CNF formula F produces a $1/2$-RR distribution that refutes $g(F)$.

Suppose that $1/2$-RR can be polynomially simulated by a refutation system given by a polynomial time binary relation R and let f be the simulation. Then we can test whether F is satisfiable by computing $R(f(h(F), g(F)), g(F))$. ◀

3 Lower bounds for the bit pigeonhole principle

We present the first of our three main lower bounds on RR distributions. Before going into details, we outline the basic structure that the proofs will follow.

To prove width lower bounds on a $1/2$-RR distribution for a CNF F, we use Proposition 5 to convert the distribution into a $(1/2, \mathcal{R})$-RR refutation (B, Π) with respect to a distribution $\mathcal{R}$ on partial assignments (we will use the terms "restriction" and "partial assignment" interchangeably). The crucial thing is to choose the distribution $\mathcal{R}$ carefully.

The ideal would be that there are many restrictions ρ from $\mathcal{R}$ which make the auxiliary formula B true, thus making it vanish and leaving us with a resolution refutation for which we already have a lower bound. To this end we use a sort of rudimentary version of the switching lemma, which we call a *fixing lemma* (a different lemma in each case, because it depends on the formula F). Intuitively this shows that, with reasonably high probability, ρ fixes the value of B to either true or false. From the definition of a $(1/2, \mathcal{R})$-RR refutation we know that $B^\rho = 0$ with probability at most a half, so we can conclude that many restrictions ρ make B true.

However, in practice it is not possible to achieve the ideal that ρ makes B true. Instead we only ask that the restricted formula B^ρ cannot be falsified by any "legal" extension $\sigma \supseteq \rho$. What counts as a legal extension depends on F – for example, for the pigeonhole principle it will be a partial assignment that encodes a matching. The definition is chosen so so that we can both prove the fixing lemma and then prove a width lower bound on Π by an adversary argument, in which the adversary only works with legal extensions of ρ.

The proof of a fixing lemma should, in principle, be a special case of a proof of a switching lemma, since we are essentially switching a CNF to a decision tree of height 0, or to a trivial DNF. However in the one case we consider in which a switching lemma is known, for the (non-bit) pigeonhole principle, we do not use it directly, but rather prove our own fixing lemma. One reason is that the usual lemma works with *syntactic* transformations of formulas and does not seem to guarantee that our *semantic* condition on B, that B is satisfied with high probability, is preserved. For the CPLS formula in the next section, there is unlikely to be any traditional switching lemma. This is because, understood very broadly, such a lemma would imply strong size lower bounds on CPLS in constant depth Frege, while we know that CPLS already has polynomial size refutations in resolution.

We continue with our lower bound proof for the bit pigeonhole principle. Let $n = 2^k$. BPHP_n is contradictory CNF asserting that a function f is an injection from $[n+1]$ to $[n]$. It has variables $(f(x))_j$ for each $x < n+1$ and $j < k$, for the jth bit of the value of $f(x)$, and consists of clauses

$$f(x) \neq y \vee f(x') \neq y$$

for all $x < x' < n+1$ and all $y < n$, using the bit-graph notation described in Section 2.

In our proof, we will only consider partial assignments in which, for every x, either all or none of the variables $(f(x))_j$ are set. We identify such assignments with the corresponding partial functions from $n+1$ pigeons to n holes.

Given a probability p, define the distribution $\mathcal{R}_p$ of partial injections ρ from $[n+1]$ into $[n]$ as follows: choose the domain of ρ by putting each pigeon into the domain independently at random with probability $1-p$, then choose uniformly at random from all possible partial injections with this domain (if all $n+1$ pigeons get put into the domain, we just take ρ to be empty). For the rest of the proof, set $p = n^{-2/3}$ and $w = n^{1/4}$.

▶ **Lemma 11** (Fixing Lemma). *Let n be sufficiently large. Suppose B is a w-CNF such that $\Pr[B^\rho = 0] \leq 1/2$. Then*

$$\Pr[\textit{there exists a partial injection } \sigma \supseteq \rho \textit{ with } B^\sigma = 0] \leq 3/4.$$

Proof. Let S be the set of $\rho \in \mathcal{R}_p$ for which there exists a partial injection $\sigma \supseteq \rho$ which falsifies B. Partition S into the set S^0 of restrictions which falsify B, and the set S^1 of restrictions which do not falsify B themselves, but which have an extension to a partial injection which falsifies B. We know $\Pr[S^0] \leq 1/2$, so it remains to bound the size of S^1.

Consider any $\rho \in S^1$. No clause in B is falsified by ρ, but there must be at least one clause which is falsified in some partial injection $\sigma \supseteq \rho$. Let C be the first such clause and let σ be such an extension of ρ falsifying it. The literals in C appear in some fixed order. Let x be the first pigeon mentioned in C which is not in the domain of ρ, and let $i < w$ be the position in C at which the first variable from pigeon x appears. Let σ' be σ restricted to the pigeons in the domain of ρ together with pigeon x, that is, $\sigma' = \rho \cup \{\langle x, \sigma(x)\rangle\}$.

Define a function θ on S^1 by $\theta : \rho \mapsto (\sigma', i)$, where σ' and i are chosen as above. Then θ is an injection, because we can first recover C from $\theta(\rho)$ as the first clause of B which is falsified in some extension of σ' to a partial injection; then we can recover x as the pigeon associated with the variable at position i in C; and finally we can recover ρ from σ' by unsetting pigeon x.

If a restriction ρ sets $m > 0$ pigeons, then the probability of ρ is

$$\Pr[\rho] = (1-p)^m p^{n+1-m} \frac{(n-m)!}{n!}.$$

Hence $\Pr[\sigma']/\Pr[\rho] = (1-p)/p(n-m)$. By the Chernoff bound, the number $n-m$ of unset pigeons is smaller than $2pn$ with exponentially high probability in n. Let S_{bad} be the set of restrictions for which this bound fails, so that $\Pr[\sigma']/\Pr[\rho] > (1-p)/2p^2n > 1/4p^2n$ for $\rho \in S^1 \setminus S_{\text{bad}}$. Partition $S^1 \setminus S_{\text{bad}}$ into subsets $S_0, \ldots, S_{w-1}$ according to the second component i of θ. On each S_i, the first component θ_1 of θ is an injection from $\mathcal{R}_p$ to $\mathcal{R}_p$ which increases probability by at least $1/4p^2n$. Therefore

$$\Pr[\theta_1[S_i]] = \sum_{\rho \in S_i} \Pr[\theta_1(\rho)] > \frac{1}{4p^2n} \sum_{\rho \in S_i} \Pr[\rho] = \frac{1}{4p^2n} \Pr[S_i].$$

Since $\Pr[\theta_1[S_i]] \leq 1$ we can conclude that $\Pr[S_i] < 4p^2n$, and hence that $\Pr[S^1 \setminus S_{\text{bad}}] < 4p^2nw = 4n^{-1/12}$. Since $\Pr[S_{\text{bad}}]$ is also exponentially small, the result follows. ◀

▶ **Theorem 12.** BPHP_n *has no $1/2$-RR distribution of width $w = n^{1/4}$.*

Proof. We will show that BPHP_n has no $(1/2, \mathcal{R}_p)$-RR refutation with this width. Suppose for a contradiction that there is such a refutation (B, Π), where B is the auxiliary w-CNF which is false in $\mathcal{R}_p$ with probability at most $1/2$.

By Lemma 11, for a random $\rho \in \mathcal{R}_p$ with probability at least $1/4$ there is no extension of ρ to a partial injection which falsifies any clause from B. Thus by the Chernoff bound we can fix one such restriction ρ which also leaves at least $pn/2 = n^{1/3}/2$ holes free.

Now consider any clause C in the refutation Π. Suppose we have a partial injection $\sigma \supseteq \rho$ that falsifies C, and suppose that C is derived by resolution from clauses $D \vee v$ and $E \vee \neg v$, where v is a variable $(f(x))_j$ for some $x < n+1$ and $j < k$. Since $|C| \leq n^{1/4}$ we can find $\sigma' \subseteq \sigma$ that falsifies C and sets at most $n^{1/4}$ pigeons not set in ρ. Hence we can find a free hole to assign to pigeon x, thus extending σ' to a partial injection which falsifies either $D \vee v$ or $E \vee \neg v$.

In this way, working inductively up through the refutation, we can find a partial injection $\sigma \supseteq \rho$ which falsifies some initial clause. But this is a contradiction, since a partial injection cannot falsify any clause from BPHP_n, and by our choice of ρ a partial injection extending ρ cannot falsify any clause from B. ◀

We show size lower bounds by combining the argument of Theorem 12 with a standard application of random restrictions to remove clauses mentioning many pigeons from Π.

▶ **Theorem 13.** BPHP_n *has no* $1/2$*-RR distribution of subexponential size, that is, of size less than* 2^{n^ε} *for some* $\varepsilon > 0$.

4 Main lower bounds and separations

4.1 A separation of resolution from narrow RR

The *coloured polynomial local search* principle (CPLS) was introduced in [12]. The propositional version of it was studied in [16]. We refer to those two papers for more on the principle, and only remark here that it is a good candidate for proving separations of this kind because it is in some sense "complete" among narrow CNFs with short resolution refutations [12], while at the same time its combinatorial structure is simple enough that we are able to come up with useful random restrictions. We take our definitions from [16].

Consider a leveled directed graph whose nodes consist of all pairs (i, x) from $[a] \times [b]$. We refer to (i, x) as *node* x *on level* i. If $i < a - 1$, this node has a single neighbour in the graph, node $f_i(x)$ on level $i+1$. Every node in the graph is coloured with some set of colours from $[c]$. CPLS expresses that the following three sentences cannot all be true at once.

1. Node 0 on level 0 has no colours.
2. For every node x on every level $i < a - 1$, if the neighbour $f_i(x)$ of x on level $i+1$ has any colour y, then x also has colour y.
3. Every node x on the bottom level $a - 1$ has at least one colour, $u(x)$.

We will express this principle as a family of propositional contradictions. Let a be any natural number and let b and c be powers of two. We will define a CNF formula $\mathrm{CPLS}_{a,b,c}$, in the following propositional variables.

- For each $i < a$, $x < b$ and $y < c$, there is a variable $G_i(x, y)$, expressing whether colour y is present at node (i, x).
- For each $i < a$, $x < b$ and $j < \log b$, there is a variable $(f_i(x))_j$, standing for the jth bit of the value of $f_i(x)$.
- For each $x < b$ and $j < \log c$, there is a variable $(u(x))_j$, standing for the jth bit of the value of $u(x)$.

▶ **Definition 14.** The formula $\mathrm{CPLS}_{a,b,c}$ consists of the following three sets of clauses, which we will call Axioms 1, 2 and 3:

Axiom 1. For each $y < c$, the clause $\neg G_0(0, y)$.

Axiom 2. For each $i < a - 1$, each pair $x, x' < b$ and each $y < c$, the clause

$$f_i(x) = x' \wedge G_{i+1}(x', y) \to G_i(x, y)\,.$$

Axiom 3. For each $x < b$ and each $y < c$, the clause

$$u(x) = y \to G_{a-1}(x, y)\,.$$

▶ **Proposition 15.** $\mathrm{CPLS}_{a,b,c}$ *has polynomial size resolution refutations.*

Proof. For $i < a$, let M_i be the set of clauses $\{\bigvee_{y<c} G_i(x, y) : x < b\}$ expressing that every node at level i has a colour. We can derive M_{a-1} from Axiom 3. Then repeatedly using Axiom 2 we can derive M_{a-2}, M_{a-3}, etc. Once we have M_0 we can derive a contradiction from Axiom 1. For more detail see [16]. ◀

▶ **Theorem 16.** *For all sufficiently large n, the formula* $\mathrm{CPLS}_{n,n,\lfloor n^{1/7}\rfloor}$ *does not have a* $1/2$*-RR distribution of width* $n^{1/8}$.

4.2 A separation of constant-depth Frege from RR

We exhibit a narrow CNF which requires exponential size 1/2-RR distributions but which, unlike the pigeonhole principle, has polynomial size refutations in constant depth Frege, in fact in Res(2). Here Res(2) is an extension of resolution in which clauses may contain conjunctions of pairs of literals (see [10]).

The formula is CPLS^2, a variant of CPLS. For each i, x, y, instead of the single variable $G_i(x, y)$ it has two variables $G_i^0(x, y)$ and $G_i^1(x, y)$. To express that colour y is present at node (i, x) we now use the conjunction $G_i^0(x, y) \wedge G_i^1(x, y)$.

As before, the formula expresses that node $(0, 0)$ has no colours; that every colour present at node $(i + 1, f_i(x))$ is also present at node (i, x); and that colour $u(x)$ is present at node $(a - 1, x)$.

▶ **Proposition 17.** $\mathrm{CPLS}^2_{a,b,c}$ *has polynomial size Res(2) refutations.*

▶ **Theorem 18.** *For all sufficiently large n, the formula* $\mathrm{CPLS}^2_{n,n,\lfloor n^{1/7}\rfloor}$ *does not have a* $1/2$*-RR distribution of size* $\leq 2^{n^{1/17}}$.

4.3 Lower bounds for the pigeonhole principle

We now consider the usual formalization of the pigeonhole principle, rather than the bit-graph version. The CNF formula PHP_n has variables p_{ij} for $i \in [n + 1]$ and $j \in [n]$ and consists of clauses $\bigvee_{j=1}^{n} p_{ij}$ for all $i \in [n + 1]$ and $\neg p_{ij} \vee \neg p_{i'j}$ for all $i, i' \in [n + 1]$ with $i \neq i'$ and all $j \in [n]$.

▶ **Theorem 19.** PHP_n *has no* $1/2$*-RR distribution of size less than* $2^{\Omega(n^{1/12})}$.

Acknowledgments. We would like to thank Pavel Hrubeš and Jan Krajíček for discussions and valuable suggestions.

References

1. Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009. doi:10.1017/CBO9780511804090.
2. Albert Atserias and Neil Thapen. The ordering principle in a fragment of approximate counting. *ACM Trans. Comput. Logic*, 15(4):29:1–11, 2014. doi:10.1145/2629555.
3. Samuel R. Buss, Leszek Aleksander Kołodziejczyk, and Neil Thapen. Fragments of approximate counting. *The Journal of Symbolic Logic*, 79(2):496–525, 2014. doi:10.1017/jsl.2013.37.
4. Mario Chiari and Jan Krajíček. Witnessing functions in bounded arithmetic and search problems. *The Journal of Symbolic Logic*, 63(03):1095–1115, 1998. doi:10.2307/2586729.
5. Vašek Chvátal and Endre Szemerédi. Many hard examples for resolution. *Journal of the ACM*, 35(4):759–768, 1988. doi:10.1145/48014.48016.
6. Stephen Cook and Robert Reckhow. The relative efficiency of propositional proof systems. *The Journal of Symbolic Logic*, 44(1):36–50, 1979. doi:10.2307/2273702.
7. Russell Impagliazzo and Jan Krajíček. A note on conservativity relations among bounded arithmetic theories. *Mathematical Logic Quarterly*, 48(3):375–377, 2002. doi:10.1002/1521-3870(200204)48:3<375::AID-MALQ375>3.0.CO;2-L.
8. Emil Jeřábek. On independence of variants of the weak pigeonhole principle. *Journal of Logic and Computation*, 17(3):587–604, 2007. doi:10.1093/logcom/exm017.
9. Jan Krajíček. Interpolation theorems, lower bounds for proof systems, and independence results for bounded arithmetic. *The Journal of Symbolic Logic*, 62(02):457–486, 1997. doi:10.2307/2275541.
10. Jan Krajíček. On the weak pigeonhole principle. *Fundamenta Mathematicae*, 170:123–140, 2001. doi:10.4064/fm170-1-8.
11. Jan Krajíček. A feasible interpolation for random resolution, 2016. Preprint arXiv:1604.06560. arXiv:1604.06560.
12. Jan Krajíček, Alan Skelley, and Neil Thapen. NP search problems in low fragments of bounded arithmetic. *The Journal of Symbolic Logic*, 72(2):649–672, 2007. doi:10.2178/jsl/1185803628.
13. Pavel Pudlák and Neil Thapen. Random resolution refutations, 2016. Electronic Colloquium on Computational Complexity, TR16-175. URL: https://eccc.weizmann.ac.il/report/2016/175/.
14. Alexander Razborov. Pseudorandom generators hard for k-DNF resolution and polynomial calculus resolution. *Annals of Mathematics*, 181(2):415–472, 2015. doi:10.4007/annals.2015.181.2.1.
15. Alan Skelley and Neil Thapen. The provably total search problems of bounded arithmetic. *Proceedings of the London Mathematical Society*, 103(1):106–138, 2011. doi:10.1112/plms/pdq044.
16. Neil Thapen. A tradeoff between length and width in resolution. *Theory of Computing*, 12(5):1–14, 2016. doi:10.4086/toc.2016.v012a005.

Graph Colouring is Hard for Algorithms Based on Hilbert's Nullstellensatz and Gröbner Bases*

Massimo Lauria[1] **and Jakob Nordström**[2]

1 **Sapienza – Università di Roma, Rome, Italy**
massimo.lauria@uniroma1.it
2 **KTH Royal Institute of Technology, Stockholm, Sweden**
jakobn@kth.se

Abstract

We consider the graph k-colouring problem encoded as a set of polynomial equations in the standard way. We prove that there are bounded-degree graphs that do not have legal k-colourings but for which the polynomial calculus proof system defined in [Clegg et al. 1996, Alekhnovich et al. 2002] requires linear degree, and hence exponential size, to establish this fact. This implies a linear degree lower bound for any algorithms based on Gröbner bases solving graph k-colouring using this encoding. The same bound applies also for the algorithm studied in a sequence of papers [De Loera et al. 2008, 2009, 2011, 2015] based on Hilbert's Nullstellensatz proofs for a slightly different encoding, thus resolving an open problem mentioned, e.g., in [De Loera et al. 2009] and [Li et al. 2016]. We obtain our results by combining the polynomial calculus degree lower bound for functional pigeonhole principle (FPHP) formulas over bounded-degree bipartite graphs in [Mikša and Nordström 2015] with a reduction from FPHP to k-colouring derivable by polynomial calculus in constant degree.

1998 ACM Subject Classification F.2.2 [Analysis of Algorithms and Problem Complexity] Nonnumerical Algorithms and Problems – Computations on Discrete Structures, F.1.3 [Computation by Abstract Devices] Complexity Measures and Classes – Relations Among Complexity Measures, G.2.2 [Discrete Mathematics] Graph Theory – Graph Algorithms, F.4.1 [Mathematical Logic and Formal Languages] Mathematical Logic – Computational Logic

Keywords and phrases proof complexity, Nullstellensatz, Gröbner basis, polynomial calculus, cutting planes, 3-colouring

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.2

1 Introduction

Given an undirected graph $G = (V, E)$ and a positive integer k, can the vertices $v \in V$ be coloured with at most k colours so that no fvertices connected by an edge have the same colour? This *graph colouring problem* is perhaps one of the most extensively studied NP-complete problems. It is widely believed that any algorithm for this problem has to run in exponential time in the worst case, and indeed the currently fastest algorithm for

* Part of this research was done while the first author was at KTH Royal Institute of Technology funded by the European Research Council (ERC) under the European Union's Seventh Framework Programme (FP7/2007–2013) / ERC grant agreement no. 279611. Later work at Universitat Politècnica de Catalunya for this project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement ERC-2014-CoG 648276 AUTAR). The second author was supported by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007–2013) / ERC grant agreement no. 279611 as well as by Swedish Research Council grants 621-2010-4797, 621-2012-5645, and 2016-00782.

3-colouring runs in time $O(1.3289^n)$ [8]. A survey on various algorithms and techniques for so-called exact algorithms is [26].

Many graph colouring instances of interest might not exhibit worst-case behaviour, however, and therefore it makes sense to study algorithms without worst-case guarantees and examine how they perform in practice. Dually, it can be of interest to study weak models of computation, which are nevertheless strong enough to capture the power of such algorithms, and prove unconditional lower bounds for these models. Obtaining such lower bounds is the goal of this work.

1.1 Brief Background

Since current state-of-the-art algorithms for propositional satisfiability such as *conflict-driven clause learning (CDCL)* [4, 32, 38] are ultimately based on *resolution* [11], it is perhaps not so surprising that this approach can be used to solve colouring problems as well. According to [6], McDiarmid developed a method for deciding k-colourability that captures many concrete algorithms [35]. This method, viewed as a proof system, is simulated by resolution.

There are exponential lower bounds for resolution proofs of non-k-colourability that apply to any such method. In particular, [6] presents average-case exponential lower bounds for random graph k-colouring instances sampled so that the graphs are highly likely not to be k-colourable. This ultimately boils down to proving width lower bounds, i.e., lower bounds on the size of a largest clause in any resolution refutation of the formula, and then using that linear width lower bounds implies exponential size lower bounds [10].

Another possible approach is to attack the k-colouring problem using algebra. Various algebraic methods have been considered in [3, 31, 33, 34]. The thesis [5] contains the first explicit attempt we know of to encode the 3-colouring problem using Hilbert's Nullstellensatz. At a high level, the idea is to write the problem as a set of polynomial equations $\{f_i(x_1, \ldots, x_n) = 0 \mid i \in [m]\}$ over a suitable field $\mathbb{F}$ so that legal colourings correspond to solutions, and if this is done in the right way it holds that this system of equations has no solution if and only if there are polynomials $g_1, \ldots, g_m$ such that $\sum_{i=1}^{m} g_i f_i = 1$. This latter equality is referred to as a *Nullstellensatz certificate* of non-colourability, and the *degree* of this certificate is the largest degree of any polynomial $g_i f_i$ in the sum. Later papers based on Nullstellensatz and Gröbner bases such as [17, 25, 37] have attracted a fair amount of attention. For this work, we are particularly interested in the sequence of papers [19, 21, 20, 18], which uses an encoding of the k-colouring problem that will be discussed more in detail later in the paper.

There seem to be no formally proven lower bounds for these algebraic methods. On the contrary, the authors of [21] report that essentially all of the benchmarks they have studied have Nullstellensatz certificates of constant (and very small) degree. Indeed, no lower bounds for graph colouring is known for the corresponding proof systems *Nullstellensatz* [7] or the stronger system *polynomial calculus* [1, 15]. Intriguingly, in a close parallel to the case for resolution it is known that strong enough lower bounds on polynomial calculus degree imply exponential lower bounds on proof size [27], but the techniques for proving degree lower bounds are much less developed than the width lower bound techniques for resolution.

Even if there are no know degree lower bounds for graph colouring, a sequence of such results exists for other formulas, although in most cases these formula are obviously false and do not express any hard computational problem. In some cases, degree lower bounds can be obtained by making an affine transformation from $\{0, 1\}$-valued variables to $\{-1, +1\}$-valued variables [9, 13], but this only works for polynomial equations with the right structure and only for fields of characteristic distinct from 2. A general and powerful method, which is

independent of the field characteristic, was developed in [2], but has turned out to be not so easy to to apply (except in a few papers such as [22, 23]). A slightly different, and in some aspects more general, version of the approach in [2] was recently presented in [36], which also highlighted the similarities and differences between resolution width lower bound techniques and polynomial calculus degree lower bound techniques. This new framework yielded a new degree lower bound which plays a key role in our paper.

1.2 Our Contributions

We exhibit families of non-k-colourable graphs of bounded degree such that the canonical encoding of the corresponding k-colouring instances into systems of polynomial equations over $\{0,1\}$-valued variables require linear degree to be refuted in polynomial calculus.

▶ **Theorem 1.1** (informal). *For any constant $k \geq 3$ there are explicit families of graphs $\{G_n\}_{n\in\mathbb{N}}$ of size $\mathrm{O}(n)$ and constant vertex degree, which are not k-colourable but for which the polynomial calculus proof system requires linear degree, and hence exponential size, to prove this fact, regardless of the underlying field.*

Our degree lower bound also applies to a slightly different encoding with primitive kth roots of unity used in [19, 20] to build k-colouring algorithms based on Hilbert's Nullstellensatz. These algorithms construct certificates of non-k-colourability by solving linear systems of equations over the coefficients of all monomials up to a certain degree.

Just as the algorithms in [19, 20], our lower bound does not work for all fields (the field must have an extension field in which there is a primitive kth root of unity). For simplicity, we state below a concrete result for Nullstellensatz certificates over GF(2) for non-3-colourability, which is one of the main cases considered in [19, 20]. We remark that this answers an open question raised in, for example, [21, 30].

▶ **Corollary 1.2.** *There are explicit families of non-3-colourable graphs such that the algorithms based on Hilbert's Nullstellensatz over* GF(2) *in* [19, 20] *need to find certificates of linear degree, and hence must solve systems of linear equations of exponential size, in order to certify non-3-colourability.*

Finally, we want to mention that the graph colouring instances that we construct turn out to be easy for the proof system *cutting planes* [16], which formalizes the integer linear programming algorithm in [14, 24] and underlies so-called *pseudo-Boolean* SAT solvers such as, for instance, *Sat4j* [29, 40].

▶ **Proposition 1.3.** *The graph colouring instances for the non-k-colourable graphs in Theorem 1.1 have polynomial-size refutations in the cutting planes proof system.*

1.3 Techniques

Perhaps somewhat surprisingly, no heavy-duty machinery is required to establish Theorem 1.1. Instead, all that is needed is a nifty reduction. Our starting point is the so-called *functional pigeonhole principle (FPHP) formula* restricted to a bipartite graph of bounded left degree k. This formula expresses the claim that a set of pigeons $i \in I$ can be mapped to a set of pigeonholes $j \in J$ in a one-to-one fashion, where in addition the pigeons are constrained so that every pigeon can choose not between all available holes but only between a set of k holes as specified by the bipartite graph. Clearly, FPHP formulas are unsatisfiable when $|I| > |J|$.

Any instance of a graph FPHP formula can be viewed as a constraint satisfaction problem by ordering the available holes for every pigeon in some arbitrary but fixed way, and then

keeping track of where each pigeon is mapped by recording the ordinal number of its chosen pigeonhole. If the cth hole for pigeon i and the c'th hole for pigeon i' is one and the same hole j, then pigeons i and i' cannot be allowed to make choices c and c' simultaneously. If we view this constraint as an edge in graph with the pigeons I as vertices, this is already close to a graph colouring instance, except that what is forbidden for the neighbours i and i' is not the same colour c but some arbitrary pair of possibly distinct colours (c, c'). However, the idea outlined above can be turned into a proper reduction from graph FPHP formulas to k-colouring instances by using appropriately constructed gadgets of constant size.

We then combine this reduction with the recent polynomial calculus degree lower bound in [36], which works as long as the underlying bipartite graph is a *boundary expander* (a.k.a. *unique-neighbour expander*). More precisely, we show that the reduction from FPHP to graph k-colouring sketched above can be computed in polynomial calculus in low degree. Therefore, any low-degree polynomial calculus refutations of the graph k-colouring instances could be used to obtain low-degree refutations of FPHP instances, but [36] tells us that FPHP instances over expander graphs require linear degree.

In order to obtain Corollary 1.2, we assume that we have a low-degree Nullstellensatz certificate (or, more generally, a polynomial calculus proof) of non-colourability for the roots-of-unity encoding in [19, 20]. Then it is not hard to show that if the field we are working in contains a primitive kth root of unity, we can apply a linear variable substitution to obtain a polynomial calculus refutation in essentially the same degree of the colouring instance in the encoding with $\{0, 1\}$-valued variables. The corollary now follows from Theorem 1.1.

As should be clear from the discussion above, the hardness of our graph colouring instances ultimately derives from the pigeonhole principle. However, this combinatorial principle is well-known to be easy for cutting planes. We establish Proposition 1.3 by showing that cutting planes can unpack the reduction described above to recover the original pigeonhole principle instance, after which this instance can be efficiently refuted.

1.4 Outline of This Paper

The rest of this paper is organized as follows. We start by presenting some proof complexity preliminaries and discussing how to encode the graph colouring problem in Section 2. In Section 3 we describe our graph k-colouring instances and prove that they are hard for polynomial calculus, and in Section 4 we show that the same instances are easy for cutting planes. We conclude in Section 5 by discussing some directions for future research. We refer to the upcoming full-length version for all missing proofs.

2 Preliminaries

Throughout this paper $x_1, \ldots, x_n$ denote $\{0, 1\}$-valued variables, where we think of 1 as true and 0 as false. We write $\mathbb{N} = \{0, 1, 2, \ldots\}$ for the natural numbers and denote $\mathbb{N}^+ = \mathbb{N} \setminus \{0\}$. For $n \in \mathbb{N}^+$ we use the standard notation $[n] = \{1, 2, \ldots, n\}$. For a set E, we use the shorthand $e \neq e' \in E$ to index over pairs of distinct elements $e, e' \in E$, $e \neq e'$.

2.1 Proof Complexity

Polynomial calculus (PC) [15] is a proof system based on algebraic reasoning where one expresses constraints over Boolean variables as polynomial equations and applies algebraic manipulations to deduce new equations. The constraints are over $\{0, 1\}$-valued variables $x_1, \ldots, x_n$, and each constraint is encoded as a polynomial Q in the ring $\mathbb{F}[x_1, \ldots, x_n]$, where

$\mathbb{F}$ is some fixed field. The intended meaning is that $Q = 0$ if and only if the constraint is satisfied, but we omit "$= 0$" below and only write the polynomial Q. A *PC derivation* of a polynomial R from a set of polynomials $\mathcal{S} = \{Q_1, \ldots, Q_m\}$ is a sequence $(P_1, \ldots, P_\tau)$ such that $P_\tau = R$ and for $1 \leq t \leq \tau$ the polynomial P_t is obtained by one of the following derivation rules:

- **Boolean axiom:** P_t is $x^2 - x$ for some variable x;
- **Initial axiom:** P_t is one of the polynomials $Q_j \in \mathcal{S}$;
- **Linear combination:** $P_t = \alpha P_i + \beta P_j$ for $1 \leq i, j < t$ and some $\alpha, \beta \in \mathbb{F}$;
- **Multiplication:** $P_t = xP_i$ for $1 \leq i < t$ and some variable x.

A *PC refutation* of $\mathcal{S}$ is a derivation of the multiplicative identity 1 of $\mathbb{F}$ from $\mathcal{S}$. Note that the Boolean axioms make sure that variables can only take values 0 and 1. For this reason, we can assume without loss of generality that all polynomials appearing in PC derivations are multilinear.

The *size* of a polynomial P is the number of distinct monomials in it when it is expanded out as a linear combination of monomials,[1] and the *degree* of P is the largest (total) degree of any monomial in P. The size of a PC derivation π is the sum of the sizes of all polynomials in π, and the degree is the maximal degree of any polynomial in π. One can also define the *length* of a PC derivation as the number of derivation steps in it, but this not so interesting a measure since it may fail to take account of polynomials of exponential size.[2] A fundamental fact about PC is that the size and degree measures are tightly related as stated next.

▶ **Theorem 2.1** ([27]). *For any set $\mathcal{S}$ of inconsistent polynomials of degree at most d' over n variables it holds that if the minimum degree of any PC refutation for $\mathcal{S}$ is at least d, then any PC refutation of $\mathcal{S}$ has size* $\exp\big(\Omega\big((d-d')^2/n\big)\big)$.

In particular, if the polynomials in $\mathcal{S}$ have constant degree but require refutations of degree linear in the number of variables n, then any refutation must have exponential size.

We remark that there is also a slightly more general version of this proof system known as *polynomial calculus (with) resolution (PCR)* [1]. The difference is that PCR has separate formal variables x and $\overline{x}$ to represent both positive and negative literals when translating CNF formulas into sets of polynomials, as well as *complementarity axioms* $x + \overline{x} - 1$ to ensure that x and $\overline{x}$ take opposite values. This yields a nicer and more well-behaved proof system. The change from PC to PCR does not affect the degree needed to refute an inconsistent set of polynomial equations, however, and Theorem 2.1 holds also for PCR. Therefore, the lower bounds we show in this paper apply both to PC and PCR.

Another aspect worth noticing is that it makes perfect sense to define polynomial calculus also for sets of polynomial equations that do not include Boolean axioms $x^2 - x$. One variant studied in the literature is to include axioms $x^k - 1$ instead, i.e., insisting that the value of x is a kth root of unity. In such a setting it is no longer necessarily true that large degree implies large space, however.

In this paper we will also consider *cutting planes (CP)* [16], which is a proof system based on manipulation of inequalities $\sum_i a_i x_i \geq \gamma$ where a_i and γ are integers and $x_1, \ldots, x_n$ are $\{0,1\}$-valued variables. A *CP derivation* of an inequality B from a set of inequalities $\mathcal{S} =$

[1] Just to make terminology precise, in this paper a *monomial* is a product of variables, a *term* is a monomial multiplied by a non-zero coefficient from the field $\mathbb{F}$, and a *polynomial* is always considered as a linear combinations of terms over pairwise distinct monomials.

[2] Indeed, if multiplication is defined to multilinearize polynomials automatically, as in, e.g., [2], then any unsatisfiable CNF formula encoded into polynomials in the natural way can be refuted in linear length – see [36] for details.

$\{A_1, \ldots, A_m\}$ is a sequence $(B_1, \ldots, B_\tau)$ such that $B_\tau = B$ and for $1 \leq t \leq \tau$ the inequality B_t is obtained by one of the following derivation rules:

- **Variable axiom:** B_t is either $x \geq 0$ or $-x \geq -1$ for some variable x.
- **Initial axiom:** B_t is some $A_j \in \mathcal{S}$;
- **Sum:** $B_t = B_i + B_j$ for $1 \leq i, j < t$.
- **Scalar multiplication:** $B_t = cB_i$ for $1 \leq i < t$ and $c \in \mathbb{N}$;
- **Division:** The inequality B_t is

$$\sum_i \frac{a_i}{c} x_i \geq \left\lceil \frac{\gamma}{c} \right\rceil \tag{1}$$

where c divides all $a_1, \ldots, a_n$ and $\sum_i a_i x_i \geq \gamma$ is some inequality B_i for $1 \leq i < t$.

A CP refutation of $\mathcal{S} = \{A_1, \ldots, A_m\}$ is a derivation from $\mathcal{S}$ of the inequality $0 \geq 1$. In what follows, we will often write $\sum_i a_i x_i \leq \gamma$ as an alias for $\sum_i -a_i x_i \geq -\gamma$, and we will also use $\sum_i a_i x_i = \gamma$ as a shorthand for the two inequalities $\sum_i a_i x_i \leq \gamma$ and $\sum_i a_i x_i \geq \gamma$.

The *length* of a CP derivation is the number of derivation steps in it. The *size* of a linear inequality $\sum_i a_i x_i \geq \gamma$ is the number of variables plus the bit size of representations of the constant term γ and all coefficients a_i, and the size of a CP derivation π is the sum of the sizes of all inequalities in π. We do not know of any degree-like measure for CP that would yield relation as that between size and degree for PC in Theorem 2.1. One usually does not distinguish too carefully between length and size for CP since by [12] all coefficients in a CP refutation can be assumed to have at most exponential size, and are hence representable with a linear number of bits.

For a partial mapping $\rho : D \to R$ from a domain D to a range R we let $\mathrm{dom}(\rho)$ denote the set of element with an image. For $d \in D \setminus \mathrm{dom}(\rho)$ we write $\rho(d) = *$. Given a partial assignment or *restriction* ρ of variables $x_1, \ldots, x_n$ to values in $\{0, 1\}$ and a polynomial P or a linear inequality A, we denote by $P\restriction_\rho$ and $A\restriction_\rho$ the polynomial and linear inequality obtained from P and A by restricting the variables in the domain of ρ to the corresponding values and making obvious syntactic simplifications. Given a derivation π in PC or CP, we denote by $\pi\restriction_\rho$ the sequence of restricted polynomials or linear inequalities, respectively. It is straightforward to verify that if π is a CP derivation of an inequality A from $\mathcal{S}$, then $\pi\restriction_\rho$ can be viewed (after simple syntactic manipulations) as a derivation of $A\restriction_\rho$ from $\mathcal{S}\restriction_\rho$ of at most the same length, and the same holds for PC with respect to size and degree.

2.2 The Graph Colouring Problem

A *legal k-colouring* of an undirected graph $G = (V, E)$ with vertices $V(G) = V$ and edges $E(G) = E$ is a mapping $\chi : V \to [k]$ such that for every edge $(u, v) \in E$ it holds that $\chi(u) \neq \chi(v)$. The *chromatic number* $\chi(G)$ of G is the smallest k such that a legal k-colouring of G exists. In the rest of this paper, colourings will often be assumed to be legal unless specified otherwise, so we will sometimes omit this prefix when no misunderstanding can occur. Also, it will sometimes be convenient to number the k colours $0, 1, \ldots, k-1$ instead of $1, 2, \ldots, k$, and we will be fairly relaxed about this issue, implicitly identifying colours 0 and k whenever convenient.

Given a graph G we can encode the k-colourability problem in a natural way as a system of polynomial equations over Boolean variables

$$\sum_{j=1}^{k} x_{v,j} = 1 \qquad v \in V(G), \tag{2a}$$

$$x_{v,j} x_{v,j'} = 0 \qquad v \in V(G),\ j \neq j' \in [k], \tag{2b}$$

$$x_{u,j} x_{v,j} = 0 \qquad (u, v) \in E(G),\ j \in [k], \tag{2c}$$

with the intended meaning that $x_{v,j} = 1$ if vertex v has colour $\chi(v) = j$. It is clear that this system of equations has a solution if and only if the graph G is k-colourable.

We will also be interested in an alternative algebraic representation of the k-colouring problem appearing, e.g., in [19, 20, 21]. In this encoding every vertex $v \in V$ has a single associated variable y_v which takes values in $\{1, \omega, \omega^2, \ldots, \omega^{k-1}\}$, where ω is a primitive kth root of unity. The intended meaning is that $y_v = \omega^j$ if vertex v has colour $j \in \{0, 1, \ldots, k-1\}$. The colouring constraints are enforced by the polynomial equations

$$y_v^k = 1 \qquad v \in V(G), \tag{3a}$$

$$\sum_{j=0}^{k-1} (y_u)^j (y_v)^{k-1-j} = 0 \qquad (u,v) \in E(G), \tag{3b}$$

where the polynomials live in a polynomial ring over a field of characteristic that is not a positive number dividing k. Clearly, Equation (3a) forces the vertex v to take some colour. A moment of thought reveals that Equation (3b) correctly encodes an edge constraint: if $y_u = \omega^a$ and $y_v = \omega^b$, then the sum evaluates to $\omega^{b(k-1)} \sum_{j=0}^{k-1} \omega^{j(a-b)}$, which equals 0 when $a \neq b$ and $k\omega^{b(k-1)} \neq 0$ otherwise. The latter formulation of k-colouring only makes sense if the characteristic of the underlying field $\mathbb{F}$ is either 0 or a positive integer that does not divide k. In this case, we also know that there exists an extension field $\mathbb{E}$ of $\mathbb{F}$ that contains a primitive kth root of unity ω [28, Chapter VI.3].

A simple but important observation for us is that the choice of the polynomial encoding is not too important if we want to study how large degree is needed in polynomial calculus when proving that some graph G is not k-colourable, provided that the field we are in contains, or can be extended to contain, a primitive kth root of unity.

▶ **Proposition 2.2.** *Suppose that Equations* (3a)–(3b) *have a polynomial calculus refutation of degree d over some field $\mathbb{F}$ of characteristic that is not a positive number dividing k. Then $\mathbb{F}$ can be extended to a field $\mathbb{E}$ containing a primitive kth root of unity ω, and it holds that Equations* (2a)–(2c) *have a polynomial calculus refutation over $\mathbb{E}$ of degree* $\max\{k, d\}$.

Proof Sketch. Given any polynomial calculus refutation π of Equations (3a)–(3b), we apply the linear substitutions

$$y_v \mapsto \sum_{j=1}^{k} x_{v,j}\omega^j \tag{4}$$

to all variables in all polynomials in this refutation to obtain a new sequence of polynomials π' in variables $x_{v,j}$. All applications of the linear combination rule in π remain valid in π', and all applications of multiplication in π can be carried out in π' by a combination of multiplication and linear combination steps. The final line of the refutation, i.e., the multiplicative identity 1, is the same in π and π'. What remains to argue is that the substituted versions of the initial axioms (3a)–(3b) in π can be derived from the axioms (2a)–(2c) available to π'. We refer to the upcoming full-length version for the details. ◀

For later use, we note that we can also encode the k-colourability problem for a graph G as a system of linear inequalities

$$\sum_{j=1}^{k} x_{v,j} \geq 1 \qquad v \in V(G), \tag{5a}$$

$$x_{v,j} + x_{v,j'} \leq 1 \qquad v \in V(G),\ j \neq j' \in [k], \tag{5b}$$

$$x_{u,j} + x_{v,j} \leq 1 \qquad (u,v) \in E(G),\ j \in [k], \tag{5c}$$

in a format amenable to cutting planes reasoning.

3 Worst-Case Lower Bound for Polynomial Calculus

We now show how to explicitly construct a family of graphs which are not k-colourable but for which polynomial calculus proofs of this fact (over any field) require degree linear in the number of vertices in the graphs. We do this in three steps:

1. First, we show how to reduce instances of *functional pigeonhole principle (FPHP) formulas* defined over bipartite graphs of bounded degree to graph colouring instances so that there is a one-to-one mapping of pigeons to holes if and only if the graph is k-colourable.
2. Then we show that polynomial calculus is able to carry out this reduction in constant degree, so that a low-degree PC proof of graph non-colourability can be used to obtain a low-degree refutation of the corresponding FPHP instance.
3. Finally, we appeal to a linear lower bound on degree for refuting FPHP instances over bipartite expander graphs from [36].

Let us start by giving a precise description of our functional pigeonhole principle instances. We have a set of pigeons I which want to fly into a set of holes J, with each pigeon flying into exactly one hole in a one-to-one fashion. However, the choices of holes for the pigeons are constrained, so that pigeon i can fly only to the holes in $J(i) \subseteq J$, where we have $|J(i)| = k$. If we use variables $p_{i,j}$ to denote that pigeon i flies into hole j, we can write the constraints on such a mapping as a set of polynomial equations

$$\sum_{j \in J(i)} p_{i,j} = 1 \qquad i \in I, \tag{6a}$$

$$p_{i,j}p_{i,j'} = 0 \qquad i \in I,\ j \neq j' \in J(i). \tag{6b}$$

$$p_{i,j}p_{i',j} = 0 \qquad i \neq i' \in I,\ j \in J(i) \cap J(i'). \tag{6c}$$

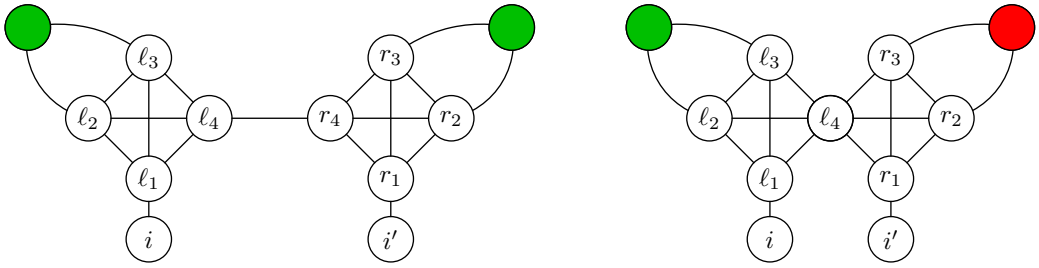
Note that an instance encoded by Equations (6a)–(6c) can also be naturally viewed as a bipartite graph B with left vertex set I, right vertex set J, and edges from each $i \in I$ to all $j \in J(i)$. In what follows, we will mostly reason about FPHP instances in terms of their representations as bipartite graphs.

In the standard setting, we let $I = [n]$ and $J = [n-1]$ for some $n \in \mathbb{N}$, in which case the collection of constraints (6a)–(6c) is clearly unsatisfiable. Nevertheless, it was shown in [36] that if the underlying bipartite graph is a so-called *boundary expander*, then any PC refutation of Equations (6a)–(6c) requires $\Omega(n)$ degree and thus, by Theorem 2.1, exponential size. For our results, we do not need to go into the technical details of this lower bound, but it suffices to use the following claim as a black box.

▶ **Theorem 3.1** ([36]). *For any integer $k \geq 3$ there is an efficiently constructible family of bipartite graphs $\{B_n\}_{n \in \mathbb{N}}$ with n vertices on the left side, $n-1$ vertices on the right side, left degree k, and right degree $O(k)$, such that any polynomial calculus refutation of the corresponding constraints* (6a)–(6c) *requires degree $\Omega(n)$.*

To be precise, the lower bound in Theorem 3.1 was proven for a slightly different encoding of Equations (6a)–(6c) – namely the one obtained from the natural translation of CNF formulas into polynomial equations – but the two encodings imply each other and can be used to derive each other in degree $O(k)$ by the implicational completeness of polynomial calculus. Hence, the lower bound holds for both encodings.

We proceed to describe the reduction from functional pigeonhole principle instances to graph colouring instances. Our starting point is an FPHP instance on a bipartite graph B with pigeons $I = [n]$ and holes J where every pigeon has exactly $d_I = k$ holes to choose from



(a) Forbidding $i \leftarrow c$ and $i' \leftarrow c$. **(b)** Forbidding $i \leftarrow c$ and $i' \leftarrow c'$ for $c \neq c'$.

Figure 1 Injectivity constraint gadgets $G_{(i,i')\not\leftarrow(c,c')}$ for $k = 4$.

and every hole can take $O(k)$ pigeons; i.e., the bipartite graph B is left-regular of degree k and has right degree $O(k)$. Based on this instance we construct a graph $G = G(B)$ such that G is k-colourable if and only if the functional pigeonhole principle on B is satisfiable.

By way of overview, the graph $G(B)$ has n special vertices corresponding to the pigeons, and the colours of these vertices encode how the pigeons are mapped to holes. For every pair of pigeons i, i' that can be mapped to the same hole j we add a gadget that forbids the colouring of the pigeon vertices i and i' that corresponds to them being mapped to hole j. These gadgets have a couple of pre-coloured vertices, but we eliminate such pre-colouring by adding one more simple gadget.

In more detail, the main idea behind the reduction is to view the choices $J(i)$ for each pigeon $i \in I$ as taking the first, second, ..., kth edge. We fix an arbitrary enumeration of the elements of $J(i)$ for each $i \in I$, associating distinct numbers $1, 2, \ldots, k$ to the edges out of the vertex i in B. We say that *pigeon i flies to hole j using its cth edge* if the edge connecting pigeon i to hole j is labelled by $c \in [k]$, and use the notation $i \leftarrow c$ for this (suppressing the information about the hole j). Pigeon i taking the cth edge corresponds to the special ith pigeon vertex being coloured with colour c.

Consider two distinct pigeons $i \neq i' \in I$ and a hole $j \in J(i) \cap J(i')$. If pigeon i flies to hole j using its cth edge and pigeon i' flies to hole j using its c'th edge, then the translation of the injectivity constraint (6c) expressed in terms of k-colourings is that vertices i and i' cannot be simultaneously coloured by colours c and c', respectively.

Let us now give a precise description of the graph gadgets we employ to enforce such injectivity constraints. These will be partially pre-coloured graphs $G_{(i,i')\not\leftarrow(c,c')}$ as depicted in Figures 1a and 1b. The gadget constructions start with two disjoint k-cliques for pigeons i and i', which we will refer to as the left and right cliques, respectively. We refer to the vertices in the left clique as $\ell_1, \ldots, \ell_k$ numbered in a clockwise fashion starting with the first vertex at the bottom, and in a symmetric fashion the vertices in the right clique are referred to as $r_1, \ldots, r_k$ numbered anti-clockwise starting at the bottom.

To the first vertex ℓ_1 in the left k-clique we connect the vertex i. To vertices $\ell_2, \ldots, \ell_{k-1}$ we connect a new vertex pre-coloured with colour c. For the right k-clique we do a similar construction: to the first vertex r_1 we connect the vertex i' and to the next $k-2$ vertices $r_2, \ldots, r_{k-1}$ we connect a new vertex pre-coloured with colour c'.

The final step of the construction depends on whether $c = c'$ or not. If $c = c'$, then we add an edge between the final two vertices ℓ_k and r_k in the cliques. If $c \neq c'$, then we instead merge these two vertices into a single vertex as shown in Figure 1b. We want to stress that except for i and i' all vertices in the construction are new vertices that do not occur in any other gadget. Let us collect for the record some properties of this gadget construction.

▶ **Claim 3.2.** *The pre-coloured graph gadget $G_{(i,i')\not\leftarrow(c,c')}$ has the following properties:*
1. *$G_{(i,i')\not\leftarrow(c,c')}$ has $\mathrm{O}(k)$ vertices.*
2. *$G_{(i,i')\not\leftarrow(c,c')}$ has two pre-coloured vertices of degree $\mathrm{O}(k)$.*
3. *For every $(b,b') \neq (c,c')$ there is a legal k-colouring χ of $G_{(i,i')\not\leftarrow(c,c')}$ extending the pre-colouring and satisfying $\chi(i) = b$ and $\chi(i') = b'$. No such legal k-colouring of $G_{(i,i')\not\leftarrow(c,c')}$ exists for $(b,b') = (c,c')$.*

Proof. The first two properties obviously hold by construction.

To prove Property 3, let us focus on the left clique in either of the two variant of the gadget. If $\chi(i) = c$, then clearly vertex ℓ_1 in the left clique cannot take colour c. Since the pre-coloured vertex connected to vertices $\ell_2, \ldots, \ell_{k-1}$ of the clique also has colour c, and since any legal colouring must use all available colours for the clique, this forces $\chi(\ell_k) = c$. If $\chi(i) \neq c$, however, then we can colour vertex ℓ_1 with colour c, and then choose any permutation of the remaining colours for the other vertices in the left clique, giving the vertex ℓ_k at least two distinct colours to choose between.

Consider now the case $c = c'$, so that we have the graph gadget $G_{(i,i')\not\leftarrow(c,c)}$ in Figure 1a. By symmetry, if $\chi(i') = c'$, then this forces $\chi(r_k) = c$, but there are at least two choices for the colour of r_k if $\chi(i') \neq c'$. It follows that if $i \leftarrow c$ and $i' \leftarrow c$, then vertices ℓ_k and r_k both have to get the same colour c to avoid conflicts in the left and right k-cliques, respectively, which causes a conflict along the edge (ℓ_k, r_k). As long as one of i and i' is assigned a colour other than c, however, $G_{(i,i')\not\leftarrow(c,c)}$ can be legally k-coloured. For $c \neq c'$ we reason analogously but use instead the graph gadget $G_{(i,i')\not\leftarrow(c,c')}$ in Figure 1b. ◀

We write $\widehat{G} = \widehat{G}(B)$ to denote the graph consisting of the union of all gadgets $G_{(i,i')\not\leftarrow(c,c')}$ for all $i \neq i' \in I$ and all c, c' such that if pigeon i uses its cth edge and pigeon i' uses its c'th edge in B, then they both end up in the same hole $j \in J$. All vertices corresponding to pigeons $i \in I$ are shared between gadgets $G_{(i,i')\not\leftarrow(c,c')}$ in $\widehat{G}$, but apart from this all subgraphs $G_{(i,i')\not\leftarrow(c,c')}$ are vertex-disjoint. We next state some properties of $\widehat{G}$.

▶ **Lemma 3.3.** *Consider an FPHP instance encoded by Equations* (6a)–(6c) *for a left-regular bipartite graph with left degree $d_I = k$ and bounded right degree $d_J = \mathrm{O}(k)$, and let $\widehat{G}$ be the partially k-coloured graph obtained as described above. Then $\widehat{G}$ has $\mathrm{O}\big(k^4|I|\big)$ vertices and maximal vertex degree $\mathrm{O}(k^2)$, and the number of pre-coloured vertices is $\mathrm{O}\big(k^2|I|\big)$. Furthermore, the partial k-colouring of $\widehat{G}$ can be extended to a complete, legal k-colouring of $\widehat{G}$ if and only if there is a way to map each pigeon $i \in I$ to some hole $j \in J$ without violating any constraint in* (6a)–(6c).

Proof. Without loss of generality we can assume that $|J| \leq k|I|$ (otherwise there are holes that cannot be used by any pigeon). Each gadget $G_{(i,i')\not\leftarrow(c,c')}$ has $\mathrm{O}(k)$ vertices and there are at most $(d_J)^2 = \mathrm{O}\big(k^2\big)$ distinct pairs of pigeons that can fly to any single hole j, meaning that we have a total of at most $\mathrm{O}\big(k^2|J|\big)$ injectivity constraint gadgets $G_{(i,i')\not\leftarrow(c,c')}$. Therefore, by a crude estimate $\widehat{G}$ has at most $\mathrm{O}\big(k^4|I|\big)$ vertices in total.

By Claim 3.2 at most $\mathrm{O}\big(k^2|I|\big)$ vertices in $\widehat{G}$ are pre-coloured. Each pigeon vertex labelled by $i \in I$ is involved in at most $d_I d_J = \mathrm{O}\big(k^2\big)$ injectivity constraint gadgets, so such vertices have degree $\mathrm{O}\big(k^2\big)$, while all other vertices have degree $\mathrm{O}(k)$.

For any complete colouring of $\widehat{G}$ extending the pre-colouring, the colours $\chi(i) = c_i$ assigned to pigeon vertices $i \in I$ define a mapping from pigeons to holes via the chosen edges c_i. It follows from Claim 3.2 that this colouring is legal only if pigeons are mapped to holes in a one-to-one fashion, which implies that Equations (6a)–(6c) are satisfiable. In the other direction, for any one-to-one mapping of pigeons to holes we can colour vertex i by the

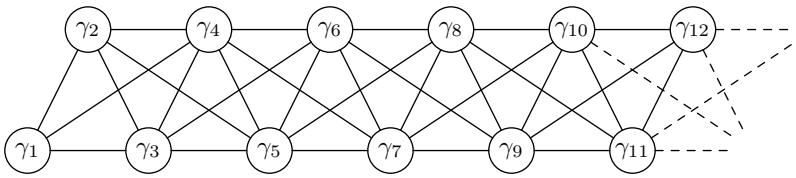


Figure 2 Pre-colouring gadget with vertices to be identified with the pre-coloured vertices in $\widehat{G}$.

colour c_i corresponding to the edge it uses to fly to its hole, and such a colouring can be combined with the pre-colouring complete, to produce a legal k-colouring. ◀

To finalize our reduction we need to get rid of the pre-coloured vertices in $\widehat{G}$. To this end, we first make the following observation. Recall that for every every pigeon $i \in I$ we fixed an enumeration of the edges to holes $j \in J(i)$ in B, so that the choice of an edge corresponds to the choice of a colour. Suppose we apply some arbitrary but fixed permutation σ on $[k]$ to all such enumerations for the pigeons $i \in I$. Clearly, this does not change the instance in any significant way. If it was the case before that pigeon i and i' could not simultaneously take the cth and c'th edges, respectively, then now these pigeons cannot simultaneously take the $\sigma(c)$th and $\sigma(c')$th edges, respectively. In other words, Lemma 3.3 is invariant with respect to any permutation of the colours $[k]$, and we could imagine the reduction as first picking some permutation σ and then constructing $\widehat{G}$ with respect to this permutation.

A simple way of achieving this effect would be to construct a separate "pre-colouring k-clique" consisting of k special vertices $\gamma_1, \ldots, \gamma_k$, and then identify all vertices in $\widehat{G}$ pre-coloured with colour c with the vertex γ_c. It is not hard to see that the resulting graph would be k-colourable if and only if the pre-colouring of $\widehat{G}$ could be extended to a complete, legal k-colouring, and using Lemma 3.3 we would obtain a valid reduction from the functional pigeonhole principle to graph k-colouring. However, the final graph would have degree $\Omega\big(k^3|I|\big)$, and we would like to obtain a graph of bounded degree.

To keep the vertex degrees independent of the size $|I|$ of the left-hand side of the FPHP bipartite graph B, we instead construct a pre-colouring gadget using a slight modification of the above idea. Consider a set $\{\gamma_1, \gamma_2, \ldots, \gamma_M\}$ of new vertices, for M to be fixed later. For every segment of k consecutive vertices $\{\gamma_t, \gamma_{t+1}, \ldots, \gamma_{t+k-1}\}$ we add all edges $\big\{(\gamma_c, \gamma_{c'}) \,\big|\, c \neq c' \in \{t, t+1, \ldots, t+k-1\}\big\}$ so that they form a k-clique as illustrated in Figure 2 (where as in Figure 1 we have $k = 4$). Next, we go through all the pre-coloured vertices in $\widehat{G}$: if a vertex should be pre-coloured by c, then we identify it with the first vertex γ_t such that $t \equiv c \pmod{k}$ and such that γ_t has not already been used at a previous step. If we choose $M = \mathrm{O}\big(k^3|I|\big)$, then we are guaranteed to have enough vertices γ_t to be able to process all pre-coloured vertices in this way.

Our final graph $G = G(B)$ is the previous graph $\widehat{G}$ with pre-coloured vertices identified with (uncoloured) vertices in the additional pre-colouring gadget as just described. Clearly, G is k-colourable if and only if the pre-colouring of $\widehat{G}$ can be completed to a legal k-colouring. We summarize the properties of our reduction in the following proposition, stated here without proof.

▶ **Proposition 3.4.** *Given a graph FPHP formula over a left-regular bipartite graph B with left degree $d_I = k$ and bounded right degree $d_J = \mathrm{O}(k)$, there is an explicit construction of a graph $G = G(B)$ such that G has $\mathrm{O}\big(k^4|I|\big)$ vertices of maximal vertex degree $\mathrm{O}(k^2)$ and is k-colourable if and only if Equations* (6a)–(6c) *are simultaneously satisfiable.*

Since our reduction encodes local injectivity constraints into local colouring constraints, it stands to reason that we should be able to translate between these two types of constraints using low degree derivations. In particular, it seems reasonable to expect that any low-degree refutation of the k-colouring problem for $G(B)$ should yield a low-degree refutation for the functional pigeohole principle on B. This is indeed the case, as stated in the next lemma.

▶ **Lemma 3.5.** *Consider the graph $G = G(B)$ obtained from a bipartite graph B as in Proposition 3.4. If the k-colourability constraints* (2a)–(2c) *for G have a PC refutation in degree d, then the functional pigeonhole principle constraints* (6a)–(6c) *defined over B have a PC refutation of degree at most* $2d$.

We will spend what remains of this section on proving this lemma. The proof is quite similar in spirit to that of Proposition 2.2. We start by assuming that we have a PC refutation of Equations (2a)–(2c) in degree d. Our first step is to substitute all variables $x_{v,j}$ in this refutation with polynomials of degree at most 2 in variables $p_{i,j}$. In the second step, we argue that if we apply this substitution to the axioms in (2a)–(2c), then we can derive the resulting substituted polynomials from Equations (6a)–(6c) by PC derivations in low degree. Taken together, this yields a PC refutation in low degree of the FPHP instance (6a)–(6c).

To describe the substitution, let us focus on a single gadget $G_{(i,i')\not\leftarrow(c,c')}$. The first step is to express all equations for this gadget as equations over variables $x_{i,1}, \dots x_{i,k}, x_{i',1}, \dots x_{i',k}$. Note that these variables are essentially the same as those from the pigeonhole principle instance, except that instead of $p_{i,j}$ we use the variable $x_{i,c}$ where c is the number of the edge pigeon i uses to fly to hole j, but for the sake of exposition we want to keep using the language of colourings.

Let w and w' be the vertices that are supposed to be pre-coloured with colours c and c', respectively. We stress that now we are considering the graph G which has no pre-coloured vertices, and in particular all the variables mentioning the vertices w and w' are unassigned. Recall that w and w' also appear in the gadget depicted in Figure 2, where they are identified with some vertices γ_t and $\gamma_{t'}$ such that $t \equiv c$ and $t' \equiv c' \pmod k$.

For any pair (b, b') of colours different from (c, c'), Claim 3.2 guarantees that we can pick some colouring $\chi_{(b,b')}$ for the gadget $G_{(i,i')\not\leftarrow(c,c')}$ such that $\chi_{(b,b')}(i) = b$, $\chi_{(b,b')}(i') = b'$, $\chi_{(b,b')}(w) = c$ and $\chi_{(b,b')}(w') = c'$. Fix for the rest of this proof such a colouring $\chi_{(b,b')}$ for the gadget $G_{(i,i')\not\leftarrow(c,c')}$ for every $(b, b') \neq (c, c')$. Then we can write the colour of any vertex v in $G_{(i,i')\not\leftarrow(c,c')}$ other than the pigeon vertices i and i' as a function of (b, b'). In more detail, we can express every variable $x_{v,j}$, for $v \notin \{i, i'\}$, as a degree-2 polynomial over the variables $x_{i,1}, \dots x_{i,k}, x_{i',1}, \dots x_{i',k}$ by summing over the monomials $x_{i,b}x_{i',b'}$ corresponding to the choices of colours (b, b') for (i, i') for which the colouring $\chi_{(b,b')}$ assigns colour j to vertex v, or in symbols

$$x_{v,j} \mapsto \sum_{(b,b')\neq(c,c'),\, \chi_{(b,b')}(v)=j} x_{i,b}x_{i',b'} \ . \tag{7}$$

Notice that for the vertices w and w' the substitutions we obtain from (7) are

$$x_{w,c} \mapsto \sum_{(b,b')\neq(c,c')} x_{i,b}x_{i',b'} \ , \tag{8a}$$

$$x_{w',c'} \mapsto \sum_{(b,b')\neq(c,c')} x_{i,b}x_{i',b'} \ , \tag{8b}$$

$$x_{w,b} \mapsto 0 \qquad (\text{for } c \neq b), \tag{8c}$$

$$x_{w',b'} \mapsto 0 \qquad (\text{for } c' \neq b'), \tag{8d}$$

since w always gets colour c and w' always gets colour c' in any colouring $\chi_{(b,b')}$.

Let us next discuss how the polynomials obtained from (2a)–(2c) after the substitution (7) can be derived in PC from (6a)–(6c). More precisely we argue that all substituted axioms can be derived from the equations

$$\sum_{b=1}^{k} x_{i,b} = 1 \ , \tag{9a}$$

$$x_{i,b}x_{i,b'} = 0 \qquad \text{(for } b \neq b'\text{)}, \tag{9b}$$

$$\sum_{b'=1}^{k} x_{i',b'} = 1 \ , \tag{9c}$$

$$x_{i',b}x_{i',b'} = 0 \qquad \text{(for } b \neq b'\text{)}, \tag{9d}$$

$$x_{i,c}x_{i',c'} = 0 \ , \tag{9e}$$

which are just the same, except for variables renaming, as the pigeon axioms (6a) and (6b) for pigeons i and i' plus the collision axiom (6c) for the hole which is the common neighbour of i and i'. In what follows we will need the equation

$$\sum_{b=1}^{k}\sum_{b'=1}^{k} x_{i,b}x_{i',b'} - x_{i,c}x_{i',c'} - 1 = 0 \tag{10}$$

which has the degree-2 proof

$$\sum_{b=1}^{k} x_{i,b}\left(\sum_{b'=1}^{k} x_{i',b'} - 1\right) + \left(\sum_{b=1}^{k} x_{i,b} - 1\right) - x_{i,c}x_{i',c'} = 0 \tag{11}$$

from (9a)–(9e).

We consider first axioms $\sum_{j=1}^{k} x_{v,j} = 1$ as in (2a) for vertices v that are not a pigeon vertex i or i'. It is straightforward to verify that such an axiom after substitution as in (7) becomes an equality on the form (10). If v is a pigeon vertex i or i', then no substitution is made and we simply keep the axiom (9a) or (9c), respectively.

Next, we consider axioms (2b) on the form $x_{v,j}x_{v,j'} = 0$, where we assume that v is not a pigeon vertex i or i' since in that case we have one of the axioms (9b) and (9d). After substitution an axiom (2b) for $v \notin \{i, i'\}$ becomes a sum of degree-4 terms of the form $x_{i,b_1}x_{i',b'_1}x_{i,b_2}x_{i',b'_2}$. Recall that the substitution associates disjoint sets of pairs (b, b') to the colours for v. Therefore, for each term $x_{i,b_1}x_{i',b'_1}x_{i,b_2}x_{i',b'_2}$ it must be that either $b_1 \neq b_2$ or $b'_1 \neq b'_2$ holds, and such a term can be derived from (9b) or (9d) by multiplication.

Let us finally consider axioms on the form $x_{u,j}x_{v,j} = 0$ for $(u, v) \in E(G)$ as in (2c). There is no edge between i and i' in our constructed graph, so for the size of the intersection between $\{u, v\}$ and $\{i, i'\}$ it holds that $0 \leq \left|\{u, v\} \cap \{i, i'\}\right| \leq 1$.

If $\left|\{u, v\} \cap \{i, i'\}\right| = 0$, then after substitution the axiom (2c) becomes a sum of degree-4 terms of the form $x_{i,b_1}x_{i',b'_1}x_{i,b_2}x_{i',b'_2}$. Consider any such term. If either $b_1 \neq b_2$ or $b'_1 \neq b'_2$, then the term can be derived from (9b) or (9d). We claim that no term can have $b_1 = b_2$ and $b'_1 = b'_2$. To see this, note that this would imply that when performing substitution as in (7) the variables $x_{u,j}$ and $x_{v,j}$ both get expanded to a sum containing $x_{i,b_1}x_{i',b'_1}$. But this would in turn mean that the colouring $\chi_{(b_1,b'_1)}$ that we fixed for the gadget $G_{(i,i')\not\sim(c,c')}$ at the start of the proof assigned colours $\chi_{(b_1,b'_1)}(u) = \chi_{(b_1,b'_1)}(v)$, which is impossible since there is an edge between u and v and $\chi_{(b_1,b'_1)}$ was chosen to be a legal colouring.

The remaining case is when we have intersection size $\left|\{u, v\} \cap \{i, i'\}\right| = 1$. Without loss of generality because of symmetry we can assume that we have an axiom $x_{u,j}x_{v,j} = 0$ for

$u \notin \{i, i'\}$ and $v = i$. The axiom becomes after substitution a sum of terms of the form $x_{i,b}x_{i',b'}x_{i,j}$. If for some term we would have $b = j$, then $\chi_{(j,b')}$ would assign the same colour j to both u and i. This is again impossible since $\chi_{(j,b')}$ is a legal colouring of the gadget by construction. Hence we have $b \neq j$ and it follows that $x_{i,b}x_{i',b'}x_{i,j}$ is derivable from (9b).

We are now almost done with the proof of Lemma 3.5. We have defined how to substitute variables $x_{v,j}$ in (2a)–(2c) and have shown that the equations that we obtain after these substitutions can be derived from Equations (6a)–(6c) in low degree. The final issue that remains it to get rid of all vertices γ_t in the pre-colouring gadget in Figure 2 that are not members of any injectivity constraint gadget $G_{(i,i')\not\leftarrow(c,c')}$. For such variables the substitution is simply an assignment: we let $x_{\gamma_t,b} \mapsto 1$ when $t \equiv b \pmod k$ and $x_{\gamma_t,b} \mapsto 0$ otherwise.[3] This immediately satisfies all axioms (2a) and (2b) for these vertices, removing these axioms from the refutation. It remains to check the axioms (2c) for any pair of connected vertices γ_t and $\gamma_{t'}$. But by construction, if γ_t and $\gamma_{t'}$ are connected it holds that $t \not\equiv t' \pmod k$. Therefore, for every $b \in [k]$ we have that either $x_{\gamma_t,b} \mapsto 0$ or $x_{\gamma_{t'},b} \mapsto 0$ holds, regardless of whether these two vertices are in some gadget $G_{(i,i')\not\leftarrow(c,c')}$ or not.

To summarize what we have done, we started with any arbitrary refutation of (2a)–(2c) and substituted all variables with degree-2 polynomials over the variables $x_{i,j}$ for $i \in [n]$. Then we proved that all these substituted axioms (and therefore the whole refutation) follow from Equations (9a)–(9c). It is straightforward to verify that, up to variable renaming, these axioms are nothing other than the FPHP axioms in (6a)–(6c). This concludes the proof of Lemma 3.5. Putting everything together, we can now state and prove our main theorem.

▶ **Theorem 3.6.** *For any integer $k \geq 3$ there is an efficiently constructible family of graphs $\{G_n\}_{n\in\mathbb{N}}$ with $\mathrm{O}(k^4 n)$ vertices of degree $\mathrm{O}(k^2)$ that do not possess k-colourings, but for which the corresponding system of polynomial equations* (2a)–(2c) *require degree $\Omega(n)$, and hence size $\exp(\Omega(n))$, to be refuted in polynomial calculus.*

Proof. Take the family of bipartite graphs $\{B_n\}_{n\in\mathbb{N}}$ as in Theorem 3.1 and apply Proposition 3.4 to this family. This yields a family of graphs $\{G_n\}_{n\in\mathbb{N}}$ as in the theorem statement. Any sublinear degree refutation for k-colouring of G_n would imply, by Lemma 3.5, a sublinear degree refutation for the functional pigeonhole principle for B_n, but this is impossible by the choice of B_n. ◀

4 Short Proofs for k-Colouring Instances in Cutting Planes

Theorem 3.6 tells us that there are non-k-colourable graphs G_n for which it is impossible for polynomial calculus to certify non-k-colourability efficiently. As is clear from our reduction, the k-colouring formulas for these graphs are essentially obfuscated instances of the functional pigeonhole principle.

It is well-known that cutting planes can easily prove that pigeonhole principle formulas are unsatisfiable by just counting the number of pigeons and holes and deduce that the pigeons are too many to fit in the holes [16]. As we show in this section, the instances of k-colouring obtained via the reduction from FPHP also have short cutting planes refutations. What these refutations do is essentially to "de-obfuscate" the k-colouring formulas to recover the original functional pigeonhole principle instances, which can then be efficiently refuted.

[3] Note that here the substitution for $x_{\gamma_t,b}$ where $t \equiv b \pmod k$ is different from the one used for vertices that are members of some gadget $G_{(i,i')\not\leftarrow(c,c')}$ in (8a) and (8b). For variables $x_{\gamma_t,b}$ where $t \not\equiv b \pmod k$ the substitution is the same as in (8c) and (8d), though.

We are going to describe our cutting planes refutation as a decision tree such that at every leaf we have a cutting planes refutation of the formula restricted by the partial assignment defined by the tree branch reaching that leaf. These refutations of the restricted versions of the formula can then be combined to yield a refutation of the original, unrestricted formula as stated in Lemma 4.1 and Proposition 4.2. The proofs of these statements are fairly routine and we omit them in this conference version of the paper.

We recall that as discussed in Section 2 we will use $\sum_i a_i x_i \leq \gamma$ as an alias for $\sum_i -a_i x_i \geq -\gamma$ and $\sum_i a_i x_i = \gamma$ as an alias for the combination of $\sum_i a_i x_i \leq \gamma$ and $\sum_i a_i x_i \geq \gamma$. In particular, we will frequently write $x = b$ for some variable x and $b \in \{0,1\}$ as a shorthand for the pair of inequalities $x \leq b$ and $-x \leq -b$.

▶ **Lemma 4.1.** *Let $b \in \{0,1\}$ and suppose that there exists a cutting planes derivation $(B_1, \ldots, B_L)$ in length L of the inequality $\sum_i a_i x_i \leq \gamma$ from the system of inequalities $\mathcal{S} \cup \{x = b\}$. Then for some $K \in \mathbb{N}$ there is a CP derivation in length $\mathrm{O}(L)$ of the inequality*

$$(-1)^{1-b} K \cdot (x - b) + \sum_i a_i x_i \leq \gamma \tag{12}$$

from $\mathcal{S}$.

▶ **Proposition 4.2.** *Let G be a graph and $k \geq 2$ be a positive integer, and let $\mathcal{S}$ be the set of inequalities* (5a)–(5c) *for G and k. If for a fixed set of vertices $u_1, u_2, \ldots, u_\ell$ in G and every choice of colours $(c_1, c_2, \ldots, c_\ell) \in [k]^\ell$ for these vertices there is a CP refutation in length at most L of the set of inequalities $\mathcal{S} \cup \{x_{u_1,c_1} = 1, x_{u_2,c_2} = 1, \ldots, x_{u_\ell,c_\ell} = 1\}$, then there is a CP refutation of $\mathcal{S}$ in length $k^{\mathrm{O}(\ell)} \cdot L$.*

We can now state the main result of this section, namely that the hard k-colouring instances for polynomial calculus constructed in Section 3 are easy for cutting planes.

▶ **Proposition 4.3.** *Let B be a left-regular bipartite graph B with left degree k and bounded right degree $\mathrm{O}(k)$, and consider the graph $G = G(B)$ in Proposition 3.4. Then if there is no complete matching of the left-hand side of B into the right-hand side, then the set of inequalities* (5a)–(5c) *encoding the k-colouring problem on G has a cutting planes refutation in length $k^{\mathrm{O}(k)} \cdot |V(B)|^{\mathrm{O}(1)}$.*

Proof Sketch. Consider the first k vertices $\gamma_1, \ldots, \gamma_k$ in the pre-colouring gadget in G as depicted in Figure 2, which form a k-clique. For every partial colouring $(c_1, c_2, \ldots, c_k) \in [k]^k$ of this k-clique we build a cutting planes refutation of

$$\mathcal{S} \cup \{x_{\gamma_1,c_1} = 1, x_{\gamma_2,c_2} = 1, \ldots, x_{\gamma_k,c_k} = 1\} \ . \tag{13}$$

The result then follows by combining all of these refutations using Proposition 4.2.

Fix a choice of colours $(c_1, c_2, \ldots, c_k) \in [k]^k$. Notice that if some colour occurs twice in this tuple, then we can derive contradiction in length $\mathrm{O}(1)$ from (13) since one of the edge axioms (5c) is violated. Suppose therefore that $(c_1, c_2, \ldots, c_k)$ is a permutation of $[k]$. We will construct a CP refutation of (13) in length $k^{\mathrm{O}(k)} \cdot |V(B)|^{\mathrm{O}(1)}$.

The system of inequalities $\mathcal{S}$ is symmetric which respect to the permutation of the colour indices, so without loss of generality we focus on giving a refutation for

$$\mathcal{S} \cup \{x_{\gamma_1,1} = 1, x_{\gamma_2,2} = 1, \ldots, x_{\gamma_k,k} = 1\} \ . \tag{14}$$

The equations $\{x_{\gamma_1,1} = 1, x_{\gamma_2,2} = 1, \ldots, x_{\gamma_k,k} = 1\}$ taken together with $\mathcal{S}$ allow us to efficiently infer $x_{\gamma_i, i \bmod k} = 1$ for all the vertices γ_i, $i \in [M]$, in the gadget in Figure 2 (where

we recall from Section 2 that we identify colours 0 and k when convenient). The resulting set of equalities and inequalities $\mathcal{S} \cup \{x_{\gamma_i, i \bmod k} = 1 \mid i \in [M]\}$ is essentially an encoding of the k-colouring problem for the partially colored graph $\widehat{G}$ in Lemma 3.3 consisting of the gadgets in Figure 1. Indeed, since the partial assignment $\{x_{\gamma_1,1} = 1, x_{\gamma_2,2} = 1, \ldots, x_{\gamma_k,k} = 1\}$ forces the colours of all vertices γ_i, $i \in [M]$, in Figure 2, this gives us back the pre-coloured vertices in the gadgets in Figure 1.

As argued in (the proof of) Lemma 3.3, $\widehat{G}$ is the union of at most $\mathrm{O}\big(k^2|V(B)|\big)$ injectivity constraint gadgets $G_{(i,i')\not\leftarrow(c,c')}$ that forbid pigeons i and i' taking their cth and c'th edges, respectively, colliding in some hole j. If we introduce the alias $p_{i,j}$ for $x_{i,c}$, where j is the hole to which the cth edge from pigeon i leads, then our goal can be described as deriving the pigeonhole axiom $p_{i,j} + p_{i',j} = x_{i,c} + x_{i',c'} \leq 1$ from the set of inequalities of the corresponding gadget $G_{(u,v)\not\leftarrow(c,c')}$. We will see shortly how to do so in length $\mathrm{O}\big(k^{\mathrm{O}(k)}\big)$. Once we extract these pigeonhole inequalities we observe that the collection of these inequalities together with the inequalities (5a) form a cutting plane encoding

$$\sum_{j \in J(i)} p_{i,j} \geq 1 \qquad i \in I, \tag{15a}$$

$$p_{i,j} + p_{i',j} \leq 0 \qquad i \neq i' \in I,\ j \in J(i) \cap J(i'). \tag{15b}$$

of the graph pigeonhole principle on the bipartite graph B with left-hand side I and right-hand side J. Such a system of inequalities has a cutting plane refutation in length $\mathrm{O}\big(|V(B)|^3\big)$ [16].

In order to derive $x_{i,c} + x_{i',c'} \leq 1$ we consider the inequalities involving vertices of $G_{(i,i')\not\leftarrow(c,c')}$ plus the equations $x_{i,c} = 1$ and $x_{i,c'} = 1$. By Claim 3.2 this is an unsatisfiable system of inequalities of size $\mathrm{O}(k)$. By the refutational completeness of cutting planes, and using Lemma 4.1 twice, we obtain a derivation of $K_1(x_{i,c} - 1) + K_2(x_{i',c'} - 1) \leq -1$ in length $\exp(\mathrm{O}(k))$. Adding multiples of axioms on the form $x - 1 \leq 0$ we get the inequality $K(x_{i,c} - 1) + K(x_{i',c'} - 1) \leq -1$ for some positive integer K, and division by K yields $x_{i,c} + x_{i',c'} \leq 1$.

We have shown how to derive contradiction is length $k^{\mathrm{O}(k)}|V(B)|^{\mathrm{O}(1)}$ for any given colouring of the vertices $\gamma_1, \ldots, \gamma_k$. We take such refutations for all k^k possible ways of assigning colours to these vertices and joint them together using Proposition 4.2 into a refutation of the original, unrestricted formula. The proposition follows. ◀

5 Concluding Remarks

In this work we exhibit explicitly constructible graphs which are non-k-colourable but which require large degree in polynomial calculus to certify this fact for the canonical encoding of the k-colouring problem into polynomial equations over $\{0, 1\}$-valued variables. This, in turn, implies that the size of any polynomial calculus proof of non-k-colourability for these graphs must be exponential measured in the number of vertices.

Our degree lower bound also applies to a slightly different encoding with primitive kth roots of unity used in [19, 20] to build k-colouring algorithms based on Hilbert's Nullstellensatz. These algorithms construct certificates of non-k-colourability by solving linear systems of equations over the coefficients of all monomials up to a certain degree. The current paper yields explicit instances for which this method needs to consider monomials up to a very large degree, and therefore has to produce a linear system of exponential size. This answers an open question raised in, e.g., [21, 30].

This leads to an important observation, however. The degree lower bound applies to both polynomial encodings discussed above, but the size lower bound only applies to the encoding

using $\{0,1\}$-valued variables. It is still conceivable that proofs of non-k-colourability in the roots of unity encoding can be small although they must have large degree. This raises the following question.

▶ **Open Problem 5.1.** *Is there a family of non-3-colourable graphs such that any polynomial calculus proof of non-3-colourability using the roots of unity encoding must require large size?*

If the answer to the question is positive, then no matter how we choose the monomials to consider for the linear system construction in [19, 20], the size of the system will have to be large.

To further reduce the size of the linear system, the algorithms in [19] make use of the symmetries in the graphs. It is a natural question how much such an approach could help for our non-k-colourable instances. It seems plausible that if we apply our construction to a randomly generated bipartite graph with appropriate parameters, then the final graph will not have many symmetries except for the local symmetries inside the gadgets. In that case our lower bound might apply for the improved version of the algorithm as well.

One serious limitation of our result is that our hard graphs are very specific, and arguably somewhat artificial. For the weaker resolution proof system an average-case exponential lower bound has been shown for Erdős–Rényi random graphs $\mathcal{G}(n,p)$ where p is slightly above the threshold value $p_k(n)$ at which the graph becomes highly likely to be non-k-colourable [6]. It is natural to ask whether these instances are hard for polynomial calculus too.

▶ **Open Problem 5.2.** *Consider a random graph sampled according to $\mathcal{G}(n,p)$ with $p > p_k(n)$, so that the graph is non-k-colourable with high probability. Does polynomial calculus require large degree to certify non-k-colourability of such graphs with high probability?*

In this paper, we also show that the graph colouring instances that are provably hard for polynomial calculus are very easy for the cutting planes proof system. We do not quite believe that graph colouring is an easy problem for cutting planes, however, and it would be interesting to find explicit candidates for hard instances for cutting planes, even if proving the actual lower bounds may be very hard. This question is also interesting for the Lasserre/Sums-of-Squares proof system. Our instances seem likely to be easy for Lasserre, since they are based on the hardness of the pigeonhole principle and this combinatorial principle is easy for Lasserre.

▶ **Open Problem 5.3.** *Find candidates for explicit hard instances of non-3-colourability for cutting planes and for Lasserre/Sums-of-squares proof systems, and then prove formally that these instances are indeed hard.*

An intriguing observation is that even though the graph colouring instances in our paper are easy for cutting planes, results from the *Pseudo-Boolean Competition 2016* indicate that they are quite hard in practice for state-of-the-art pseudo-Boolean solvers [39]. This is even more interesting considering that the cutting planes refutations that we construct have small rank (i.e., the maximum number of application of the division rules along any path in the proof graph is small).

Acknowledgements. We are grateful to Mladen Mikša and Alexander Razborov for stimulating discussions and helpful feedback during various stages of this project. We would also like to thank Jan Elffers for running experiments with pseudo-Boolean solvers on instances obtained from our reduction from functional pigeonhole principle formulas to graph colouring, demonstrating that these formulas are hard in practice. Last but not least, a big thanks to the anonymous CCC reviewers, who helped us catch some typos and bugs that really should not have been there, and whose suggestions helped improve the exposition considerably.

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 Michael Alekhnovich and Alexander A. Razborov. Lower bounds for polynomial calculus: Non-binomial case. *Proceedings of the Steklov Institute of Mathematics*, 242:18–35, 2003. Available at `http://people.cs.uchicago.edu/~razborov/files/misha.pdf`. Preliminary version in *FOCS'01*.

3 Noga Alon and Michael Tarsi. Colorings and orientations of graphs. *Combinatorica*, 12(2):125–134, June 1992.

4 Roberto J. Bayardo Jr. and Robert Schrag. Using CSP look-back techniques to solve real-world SAT instances. In *Proceedings of the 14th National Conference on Artificial Intelligence (AAAI'97)*, pages 203–208, July 1997.

5 David Allen Bayer. *The Division Algorithm and the Hilbert Scheme*. PhD thesis, Harvard University, Cambridge, MA, USA, June 1982. Available at `https://www.math.columbia.edu/~bayer/papers/Bayer-thesis.pdf`.

6 Paul Beame, Joseph C. Culberson, David G. Mitchell, and Cristopher Moore. The resolution complexity of random graph k-colorability. *Discrete Applied Mathematics*, 153(1–3):25–47, December 2005.

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 Richard Beigel and David Eppstein. 3-coloring in time $O(n^{1.3289})$. *Journal of Algorithms*, 54(2):168–204, February 2005.

9 Eli Ben-Sasson and Russell Impagliazzo. Random CNF's are hard for the polynomial calculus. *Computational Complexity*, 19:501–519, 2010. Preliminary version in *FOCS'99*.

10 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*.

11 Archie Blake. *Canonical Expressions in Boolean Algebra*. PhD thesis, University of Chicago, 1937.

12 Samuel R. Buss and Peter Clote. Cutting planes, connectivity and threshold logic. *Archive for Mathematical Logic*, 35:33–63, 1996.

13 Samuel R. Buss, Dima Grigoriev, Russell Impagliazzo, and Toniann Pitassi. Linear gaps between degrees for the polynomial calculus modulo distinct primes. *Journal of Computer and System Sciences*, 62(2):267–289, March 2001. Preliminary version in *CCC'99*.

14 Vašek Chvátal. Edmond polytopes and a hierarchy of combinatorial problems. *Discrete Mathematics*, 4(1):305–337, 1973.

15 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.

16 William Cook, Collette Rene Coullard, and György Turán. On the complexity of cutting-plane proofs. *Discrete Applied Mathematics*, 18(1):25–38, November 1987.

17 Jesús A. De Loera. Gröbner bases and graph colorings. *Beiträge zur Algebra und Geometrie*, 36(1):89–96, January 1995. Available at `https://www.emis.de/journals/BAG/vol.36/no.1/`.

18 Jesús A. De Loera, Susan Margulies, Michael Pernpeintner, Eric Riedl, David Rolnick, Gwen Spencer, Despina Stasi, and Jon Swenson. Graph-coloring ideals: Nullstellensatz certificates, Gröbner bases for chordal graphs, and hardness of Gröbner bases. In *Proceedings*

of the 40th International Symposium on Symbolic and Algebraic Computation (ISSAC'15), pages 133–140, July 2015.

19 Jesús A. De Loera, Jon Lee, Peter N. Malkin, and Susan Margulies. Hilbert's Nullstellensatz and an algorithm for proving combinatorial infeasibility. In *Proceedings of the 21st International Symposium on Symbolic and Algebraic Computation (ISSAC'08)*, pages 197–206, July 2008.

20 Jesús A. De Loera, Jon Lee, Peter N. Malkin, and Susan Margulies. Computing infeasibility certificates for combinatorial problems through Hilbert's Nullstellensatz. *Journal of Symbolic Computation*, 46(11):1260–1283, November 2011.

21 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*, 18:551–582, July 2009.

22 Nicola Galesi and Massimo Lauria. On the automatizability of polynomial calculus. *Theory of Computing Systems*, 47:491–506, August 2010.

23 Nicola Galesi and Massimo Lauria. Optimality of size-degree trade-offs for polynomial calculus. *ACM Transactions on Computational Logic*, 12:4:1–4:22, November 2010.

24 Ralph E. Gomory. An algorithm for integer solutions of linear programs. In R.L. Graves and P. Wolfe, editors, *Recent Advances in Mathematical Programming*, pages 269–302. McGraw-Hill, New York, 1963.

25 Christopher J. Hillar and Troels Windfeldt. Algebraic characterization of uniquely vertex colorable graphs. *Journal of Combinatorial Theory, Series B*, 98(2):400–414, March 2008.

26 Thore Husfeldt. Graph colouring algorithms. In Lowell W. Beineke and Robin J. Wilson, editors, *Topics in Chromatic Graph Theory*, Encyclopedia of Mathematics and its Applications, chapter 13, pages 277–303. Cambridge University Press, May 2015.

27 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.

28 Serge Lang. *Algebra*. Springer New York, 2005.

29 Daniel Le Berre and Anne Parrain. The Sat4j library, release 2.2. *Journal on Satisfiability, Boolean Modeling and Computation*, 7:59–64, 2010.

30 Bo Li, Benjamin Lowenstein, and Mohamed Omar. Low degree Nullstellensatz certificates for 3-colorability. *The Electronic Journal of Combinatorics*, 23(1), January 2016.

31 László Lovász. Stable sets and polynomials. *Discrete Mathematics*, 124(1-3):137–153, January 1994.

32 João P. Marques-Silva and Karem A. Sakallah. GRASP: A search algorithm for propositional satisfiability. *IEEE Transactions on Computers*, 48(5):506–521, May 1999. Preliminary version in *ICCAD'96*.

33 Yuri V. Matiyasevich. A criterion for vertex colorability of a graph stated in terms of edge orientations. *Diskretnyi Analiz*, 26:65–71, 1974. English translation of the Russian original. Available at `http://logic.pdmi.ras.ru/~yumat/papers/22_paper/`.

34 Yuri V. Matiyasevich. Some algebraic methods for calculating the number of colorings of a graph. *Journal of Mathematical Sciences*, 121(3):2401–2408, May 2004.

35 Colin McDiarmid. Colouring random graphs. *Annals of Operations Research*, 1(3):183–200, October 1984.

36 Mladen Mikša and Jakob Nordström. A generalized method for proving polynomial calculus degree lower bounds. In *Proceedings of the 30th Annual Computational Complexity Conference (CCC'15)*, volume 33 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 467–487, June 2015.

37 Michal Mnuk. Representing graph properties by polynomial ideals. In *Proceedings of the 4th International Workshop on Computer Algebra in Scientific Computing (CASC'01)*, pages 431–444, September 2001.

38 Matthew W. Moskewicz, Conor F. Madigan, Ying Zhao, Lintao Zhang, and Sharad Malik. Chaff: Engineering an efficient SAT solver. In *Proceedings of the 38th Design Automation Conference (DAC'01)*, pages 530–535, June 2001.

39 Pseudo-Boolean competition 2016: Results by benchmark for category no optimisation, small integers, linear constraints (DEC-SMALLINT-LIN). `http://www.cril.univ-artois.fr/PB16/results/globalbybench.php?idev=81&idcat=47`, 2016.

40 Sat4j: The Boolean satisfaction and optimization library in Java. `http://www.sat4j.org/`.

Representations of Monotone Boolean Functions by Linear Programs*

Mateus de Oliveira Oliveira[1] and Pavel Pudlák[2]

1 **University of Bergen, Bergen, Norway**
`mateus.oliveira@uib.no`
2 **Czech Academy of Sciences, Prague, Czech Republic**
`pudlak@math.cas.cz`

Abstract

We introduce the notion of monotone linear-programming circuits (MLP circuits), a model of computation for partial Boolean functions. Using this model, we prove the following results.

1. MLP circuits are superpolynomially stronger than monotone Boolean circuits.
2. MLP circuits are exponentially stronger than monotone span programs.
3. MLP circuits can be used to provide monotone feasibility interpolation theorems for Lovász-Schrijver proof systems, and for mixed Lovász-Schrijver proof systems.
4. The Lovász-Schrijver proof system cannot be polynomially simulated by the cutting planes proof system. This is the first result showing a separation between these two proof systems.

Finally, we discuss connections between the problem of proving lower bounds on the size of MLPs and the problem of proving lower bounds on extended formulations of polytopes.

1998 ACM Subject Classification F.2.2 [Nonnumerical Algorithms and Problems] Complexity of Proof Procedures

Keywords and phrases Monotone Linear Programming Circuits, Lovász-Schrijver Proof System, Cutting-Planes Proof System, Feasible Interpolation, Lower Bounds

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.3

1 Introduction

Superpolynomial lower bounds on the size of Boolean circuits computing explicit Boolean functions have only been proved for circuits from some specific families of circuits. A prominent role among these families is played by *monotone Boolean circuits*. Exponential lower bounds on monotone Boolean circuits were proved already in 1985 by Razborov [26]. In 1997 Krajíček discovered that lower bounds on monotone complexity of particular partial Boolean functions can be used to prove lower bounds on resolution proofs [18]. Incidentally, the functions used in Razborov's lower bound were just of the form needed for resolution lower bounds. Exponential lower bounds on resolution proofs had been proved before (coincidentally about at the same time as Razborov's lower bounds). Krajíček came up with a new general method, the so called *feasible interpolation,* that potentially could be used for other proof systems. Indeed, soon after his result, this method was used to prove exponential lower bounds on the cutting-planes proof system [22, 15]. That lower

* Mateus de Oliveira Oliveira acknowledges support from the ERC Advanced Grant 339691 (FEALORA) and from the Bergen Research Foundation. Pavel Pudlák acknowledges support from the ERC Advanced Grant 339691 (FEALORA).

bound is based on a generalization of Razborov's lower bounds to a more general monotone computational model, the *monotone real circuits*. Another monotone computational model for which superpolynomial lower bounds have been obtained is the *monotone span program* model [2, 11]. An exponential lower bound on the size of monotone span programs have been recently obtained in [7]. For a long time the best known lower bound for this model of computation was of the order of $n^{\Omega(\log n)}$ [2]. Again, superpolynomial lower bounds on the size of monotone span programs can be used to derive lower bounds on the degree of Nullstellensatz proofs, as shown in [23].[1]

The results listed above suggest that proving lower bounds on stronger and stronger models of monotone computation may be a promising approach towards proving lower bounds on stronger proof systems. Indeed, in his survey article [27] Razborov presents the problem of understanding feasible interpolation for stronger systems as one of the most challenging ones.

In this work we introduce several computational models based on the notion of *monotone linear program*. In particular, we introduce the notion of *monotone linear-programming gate* (MLP gate). In its most basic form, an MLP gate is a *partial* function $g : \mathbb{R} \to \mathbb{R}$ of the form $g(y) = \max\{c \cdot x \mid Ax \leq b + By, x \geq 0\}$ where y is a set of input variables, and B is a non-negative matrix. The complexity of such a gate is defined as the number of rows plus the number of columns in the matrix A. For each assignment $\alpha \in \mathbb{R}^n$ of the variables y, the value $g(\alpha)$ is the optimal value of the linear program with objective function $c \cdot x$, and constraints $Ax \leq b + B\alpha$. The requirement that $B \geq 0$ guarantees monotonicity, i.e., that $g(\alpha) \leq g(\alpha')$ whenever $g(\alpha)$ is defined and $\alpha \leq \alpha'$. We note that the value $g(\alpha)$ is considered to be undefined if the associated linear program $\max\{c \cdot x \mid Ax \leq b + B\alpha\}$ has no solution. Other variants of MLP gates are defined in a similar way by allowing the input variables to occur in the objective function, and by allowing the corresponding linear programs to be minimizing or maximizing. We say that an MLP gate is weak if the input variables occur either in the objective function or in the constraints. We say that an MLP gate is strong if the input variables occur in both the objective function and in the constraints.

An MLP circuit is a straightforward generalization of the notion of unbounded-fan-in (monotone) Boolean circuit where MLP gates are used instead of Boolean gates. In Theorem 3 we show that if all gates of an MLP circuit C are weak, then this circuit can be simulated by a single weak MLP gate ℓ_C whose size is polynomial on the size of C. Since the AND and OR gates can be faithfully simulated by weak MLP gates, we have that monotone Boolean circuits can be polynomially simulated by weak MLP circuits (Theorem 4). In contrast, we show that weak MLP gates are super-polynomially stronger than monotone Boolean circuits. On the one hand, Razborov has shown that that any monotone Boolean circuit computing the *bipartite perfect matching function* $\mathrm{BPM}_n : \{0,1\}^{n^2} \to \{0,1\}$ must have size at least $n^{\Omega(\log n)}$. On the other hand, a classical result in linear programming theory [29] can be used to show that the same function can be computed by weak MLP gates of polynomial size.

In [2], Babai, Gál and Wigderson showed that there is a function that can be computed by span programs of linear size but which requires superpolynomial-size monotone Boolean circuits. Recently, Cook et al. [7] showed that there is a function that can be computed by polynomial-size monotone Boolean circuits, but that requires exponential-size monotone span programs over the reals. Therefore, monotone span programs (which we will abbreviate by MSPs) and monotone Boolean circuits are incomparable in the sense that neither of these

[1] We note however that strong degree lower bounds for Nullstellensatz proofs can be proved using more direct methods [3, 6, 13, 1].

models can polynomially simulate the other. In Theorem 7 we show that a particular type of weak MLP gate can polynomially simulate monotone span programs over the reals. On the other hand, by combining the results in [7] with Theorem 7, we have that these weak MLP gates are exponentially stronger than monotone span programs over reals. Therefore, while monotone Boolean circuits are incomparable with MSPs, weak MLP-gates are strictly stronger than both models of computation.

Next we turn to the problem of proving a monotone interpolation theorem for Lovász-Schrijver proof systems [20]. Currently, size lower bounds for these systems have been proved only with respect to tree-like proofs [21], and therefore, it seems reasonable that a monotone interpolation theorem for this system may be a first step towards proving size lower bounds for general LS proof systems. Towards this goal we show that MLP circuits which are constituted by strong MLP gates can be used to provide a *monotone* feasible interpolation theorem for LS proof systems. In other words, we reduce the problem of proving superpolynomial lower bounds for the size of LS proofs, to the problem of proving lower bounds on the size of MLP circuits with strong gates.

While circuits with weak MLP gates can be collapsed to a single weak MLP gate, we do not know how to collapse MLP circuits with strong gates into a single strong gate. Nevertheless, in Theorem 10 we show that a single weak MLP gate suffices in a monotone interpolation theorem for *mixed LS proofs*. These are proofs in which, on top of variables representing 0s and 1s, there are also variables that range over real numbers. This interpolation theorem implies two things. First, the cutting-planes proof system cannot polynomially simulate the LS proof system (Corollary 18). Understanding the mutual relation between the power of the cutting-planes proof system and the LS proof system is a longstanding open problem in proof complexity theory. Our result solves one direction of this mutual relation by showing that for some tautologies, LS proofs can be superpolynomially more concise than cutting-planes proofs. Second, using this interpolation theorem, and a size lower bound for monotone real circuits due to Fu [10], we can show that MLP-circuits cannot be polynomially simulated by monotone real circuits (Theorem 19).

Monotone linear programs programs may be regarded as a generalization of both monotone Boolean circuits and monotone span programs. Since superpolynomial lower bounds for these two latter formalisms were proved via rather distinct formalisms, it is reasonable to expect that new lower bound methods will need to be developed in order to prove superpolynomial lower bounds for the size of weak monotone linear programs. A possible approach is to try to strength recent lower bounds obtained for the extension complexity of polytopes whose vertices correspond to minterms of certain monotone Boolean functions [28, 9, 4, 5]. To prove a lower bound on the size of weak MLP gates, it will be necessary to prove lower bounds on the size of extended formulations for all polytopes of a certain form that separate minterms from maxterms. This is clearly a harder problem than proving lower bounds on the extension complexity of a single polytope. Nevertheless, there are certain results that point in this direction [4, 5]. In any case, Theorem 19 suggests that this will not be an easy task. The theorem gives an example of a monotone function whose set of ones requires exponentially large extended formulation, but whose minterms can be separated from a large subset of maxterms by a polynomial size weak MLP gate.

In this extended abstract all proofs are omitted.

2 Monotone Linear-Programming Gates

▶ **Definition 1** (MLP Gate). A *monotone linear-programming gate*, or MLP gate, is a partial function $\ell : \mathbb{R}^n \to \mathbb{R} \cup \{*\}$ whose value at each point $y \in \mathbb{R}^n$ is specified via a monotone linear program. More precisely, we consider the following six types of MLP gates:

MAX-RIGHT: $\ell(y) = \max\{c^T \cdot x \mid Ax \leq b + By,\ x \geq 0\}$

MIN-RIGHT: $\ell(y) = \min\{c^T \cdot x \mid Ax \geq b + By,\ x \geq 0\}$

MAX-LEFT: $\ell(y) = \max\{(c + Cy)^T \cdot x \mid Ax \leq b,\ x \geq 0\}$

MIN-LEFT: $\ell(y) = \min\{(c + Cy)^T \cdot x \mid Ax \geq b,\ x \geq 0\}$

MAX: $\ell(y) = \max\{(c + Cy)^T \cdot x \mid Ax \leq b + By,\ x \geq 0\}$

MIN: $\ell(y) = \min\{(c + Cy)^T \cdot x \mid Ax \geq b + By,\ x \geq 0\}$

where A is a matrix in $\mathbb{R}^{m \times k}$, b is a vector in $\mathbb{R}^m$, c is a vector in $\mathbb{R}^k$, and B and C are nonnegative matrices in $\mathbb{R}^{m \times n}$, i.e., $B \geq 0$ and $C \geq 0$.

Intuitively, the variables y should be regarded as input variables, while the variables x should be regarded as internal variables. If the linear program specifying a gate $\ell(y)$ has no solution when setting y to a particular point $\alpha \in \mathbb{R}^n$, then we set $\ell(\alpha) = *$. In other words, in this case we regard the value $\ell(\alpha)$ as being undefined. We note that the requirement $B \geq 0, C \geq 0$ guarantees that the gates introduced above are monotone. More precisely, if $\alpha \leq \alpha'$, and both $\ell(\alpha)$ and $\ell(\alpha')$ are well defined, then $\ell(\alpha) \leq \ell(\alpha')$. The size $|\ell|$ of an MLP gate ℓ is defined as the number of rows plus the number of columns in the matrix A.

The gates of type MAX-RIGHT, MAX-LEFT, MIN-RIGHT and MIN-LEFT are called weak gates. Note that in these gates, the input variables y occur either only in the objective function, or only in the constraints. The gates of type MAX and MIN are called strong gates. The input variables in strong gates occur both in the constraints and in the objective function.

Circuits that are constituted of MLP gates are called *MLP circuits*.

▶ **Definition 2** (MLP-Circuit Representation). We say that an MLP circuit C represents a partial Boolean function $F : \{0,1\}^n \to \{0,1,*\}$ if the following conditions are satisfied for each $a \in \{0,1\}^n$.

1. $C(a) > 0$ if $F(a) = 1$.
2. $C(a) \leq 0$ if $F(a) = 0$.

We say that an MLP-circuit C *sharply* represents $F : \{0,1\}^m \to \{0,1,*\}$ if $C(a) = 1$ whenever $F(a) = 1$ and $C(a) = 0$ whenever $F(a) = 0$. We define the size of an MLP circuit C as the sum of the sizes of MLP gates occurring in C. The next theorem states that if all gates in an MLP circuit C are weak MLP gates with the same type τ, then this circuit can be polynomially simulated by a *single* MLP gate ℓ_C of type τ.

▶ **Theorem 3** (From Circuits to Gates). *Let C be an MLP circuit of size s where all gates in C are weak MLP gates of type τ. Then there is an MLP gate ℓ_C of type τ and size $O(s)$ such that for each $a \in \mathbb{R}^n$ for which $C(a)$ is defined, $\ell_C(a) = C(a)$.*

3 Weak MLP Gates vs Monotone Boolean Circuits

In this section we show that partial Boolean functions that can be represented by monotone Boolean circuits of size s can also be sharply represented by weak MLP gates of size $O(s)$. On the other hand, we exhibit a partial function that can be represented by polynomial-size max-right MLP gates, but which require Boolean circuits of superpolynomial size.

▶ **Theorem 4.** *Let $F : \{0,1\}^n \to \{0,1,*\}$ be a partial Boolean function, and let C be a Boolean circuit of size s representing F. Then for any weak type τ, F can be sharply represented by an MLP gate of type τ and size $O(s)$.*

Let $BPM_n : \{0,1\}^{n^2} \to \{0,1\}$ be the Boolean function that evaluates to 1 on an input $p \in \{0,1\}^{n^2}$ if and only if p represents a bipartite graph with a perfect matching. The next theorem, whose proof is based on a classical result in linear programming theory (Theorem 18.1 of [29]) states that the function BPM_n has small MAX-RIGHT MLP representations.

▶ **Theorem 5.** *The Boolean function $BPM_n : \{0,1\}^{n^2} \to \{0,1\}$ can be represented by a* MAX-RIGHT *MLP gate of size $n^{O(1)}$.*

In a celebrated result, Razborov proved a lower bound of $n^{\Omega(\log n)}$ for the size of monotone Boolean circuits computing the function BPM_n [26]. By combining this result with Theorem 5, we have the following corollary.

▶ **Corollary 6.** MAX-RIGHT *MLP gates cannot be polynomially simulated by monotone Boolean circuits.*

We note that the gap between the complexity of MAX-RIGHT MLP gates and the complexity of Boolean formulas computing the BPM_n function is even exponential, since Raz and Wigderson have shown a linear lower-bound on the depth of monotone Boolean circuits computing BPM_n [24].

3.1 Monotone Span Programs

Monotone span programs (MSP) were introduced by Karchmer and Wigderson [17]. Such a program, which is defined over an arbitrary field $\mathbb{F}$, is specified by a vector $c \in \mathbb{F}^k$ and a labeled matrix $A^\rho = (A, \rho)$ where A is a matrix in $\mathbb{F}^{m \times k}$, and $\rho : \{1, ..., m\} \to \{p_1, ..., p_n, *\}$ labels rows in A with variables in p_i or with the symbol $*$ (meaning that the row is unlabeled). For an assignment $p := w$, let $A^\rho_{\langle w \rangle}$ be the matrix obtained from A by deleting all rows labeled with variables which are set to 0. A span program (A^ρ, c) represents a partial Boolean function $F : \{0,1\}^n \to \{0,1,*\}$ if the following conditions are satisfied for each $w \in \{0,1\}^n$.

$$F(w) = \begin{cases} 1 \;\Rightarrow\; \exists y,\ y^T A^\rho_{\langle w \rangle} = c^T \\ 0 \;\Rightarrow\; \neg\exists y,\ y^T A^\rho_{\langle w \rangle} = c^T \end{cases} \tag{1}$$

That is, if $F(p) = 1$ then c is a linear combination of the rows of $A_{\langle w \rangle}$, while if $F(p) = 0$, then c cannot be cast as such linear combination. We define the size of a span program (A^ρ, c) as the number of rows plus the number of columns in the matrix A. The next theorem states that functions that can be represented by small MSPs over the reals can also be represented by small MIN-RIGHT MLP gates.

▶ **Theorem 7.** *Let $F : \{0,1\}^n \to \{0,1\}$ be a Boolean function. If F can be represented by an MSP of size s over the reals, then F can be represented by a* MIN-RIGHT *MLP gate of size $O(s)$.*

It has been recently shown that there is a family of functions $\mathrm{GEN}_n : \{0,1\}^n \to \{0,1\}$ which can be computed by polynomial-size monotone Boolean circuits but which require monotone span programs over the reals of size $\exp(n^{\Omega(1)})$ [7]. On the other hand, since by Theorem 4, monotone Boolean circuits can be polynomially simulated by weak MLP gates of any type, we have that weak MLP gates of size polynomial in n can represent the function $\mathrm{GEN}_n : \{0,1\}^n \to \{0,1\}$. Therefore, we have the following corollary.

▶ **Corollary 8.** *Weak MLP gates cannot be polynomially simulated by monotone span programs over the reals.*

4 Lovász-Schrijver and Cutting-Planes Proof Systems

4.1 The Lovász-Schrijver Proof System

The Lovász-Schrijver proof system is a refutation system based on the Lovász-Schrijver method for solving integer linear programs [20]. During the past two decades several variants (probably nonequivalent) of this system have been introduced. In this work we will be only concerned with the basic system LS. In Lovász-Schrijver systems the domain of variables is restricted to $\{0,1\}$, i.e., they are Boolean variables. Given an unfeasible set of inequalities Φ, the goal is to use the axioms and rules of inference defined below to show that the inequality $0 \geq 1$ is implied by Φ.

- **Axioms:**
  1. $0 \geq 0$, $1 \geq 0$, $1 \geq 1$,
  2. $0 \leq p_j \leq 1$,
  3. $p_i^2 - p_i = 0$ (integrality).
- **Rules:**
  1. *Positive linear combinations of linear and quadratic inequalities*,
  2. *Multiplication:* Given a linear inequality $\sum_i c_i p_i - d \geq 0$, and a variable p_j, derive

$$p_j(\sum_i c_i p_i - d) \geq 0 \quad \text{and} \quad (1 - p_j)(\sum_i c_i p_i - d) \geq 0.$$

Note that using multiplication we may produce quadratic inequalities, but we can only apply the multiplication rule to linear inequalities, hence all inequalities are at most quadratic. Axiom (3) corresponds to two inequalities, but it suffices to use $p_i^2 - p_i \geq 0$, since the other inequality $p_i^2 - p_i \leq 0$ follows from Axiom (2) and Rule (2). We also observe that the inequality $1 \geq 0$ can be derived from the axioms $p_i \geq 0$ and $1 - p_i \geq 0$.

A proof Π of an inequality $\sum_i c_i p_i - d \geq 0$ from Φ is a sequence of inequalities such that every inequality in the sequence is either an element of Φ or is derived from previous ones using some LS rule. We say that Π is a refutation of the set of inequalities Φ, if the last inequality is $-d \geq 0$ for some $d > 0$.

The LS proof system is implicationally complete. This means that if an inequality $\sum_i c_i p_i - d \geq 0$ is semantically implied by an initial set of inequalities Φ, then $\sum_i c_i p_i - d \geq 0$ can be derived from Φ by the application of a sequence of LS-rules [20].

Superpolynomial lower bounds on the size of LS proofs have been obtained only in the restricted case of tree-like proofs [21]. The problem of obtaining superpolynomial lower bounds for the size of DAG-like LS proofs remains a tantalizing open problem in proof complexity theory.

The LS proof system is stronger than Resolution. It can be shown that resolution proofs can be simulated by LS proofs with just a linear blow up in size. Additionally, the Pigeonhole

principle has LS proofs of polynomial size, while this principle requires exponentially long resolution proofs [14]. On the other hand, the relationship between the power of the LS proof system and other well studied proof system is still elusive. For instance, previous to this work, nothing was known about how the LS proof system relates to the cutting-planes proof system with respect to polynomial-time simulations. In Subsection 4.4 we will show that there is a family of sets of inequalities which have polynomial-size DAG-like LS refutations, but which require superpolynomial-size cutting-planes refutations. This shows that the cutting-planes proof system cannot polynomially simulate the LS proof system. The converse problem, of determining whether the LS proof system polynomially simulates the cutting-planes proof system, remains open.

4.2 Feasible interpolation

Feasible interpolation is a method that can sometimes be used to translate circuit lower bounds into lower bounds for the size of refutations of Boolean formulas and linear inequalities. Let $\Phi(p, q, r)$ be an unsatisfiable Boolean formula which is a conjunction of formulas $\Phi_1(p, q)$ and $\Phi_2(p, r)$ where q and r are disjoint sets of variables. Since $\Phi(p, q, r)$ is unsatisfiable, it must be the case that for each assignment a of the variables p, either $\Phi_1(a, q)$ or $\Phi_2(a, r)$ is unsatisfiable, or both. Given a proof Π of unsatisfiability for $\Phi(p, q, r)$, an *interpolant* is a Boolean circuit $C(p)$ such that for every assignment a to the variables p:

1. if $C(a) = 0$, then $\Phi_1(x, a)$ is unsatisfiable, and
2. if $C(a) = 1$, then $\Phi_2(y, a)$ is unsatisfiable.

If both formulas are unsatisfiable, then $C(a)$ can be either of the two values. Krajíček has shown that given a resolution refutation Π of a CNF formula, one can construct an interpolant $C(p)$ whose size is polynomial in the size of Π [18]. Krajíček 's interpolation theorem has been generalized, by himself and some other authors, to other proof systems such as the cutting-planes proof system and the Lovász-Schrijver proof system [8].

In principle, such *feasible interpolation* theorems could be used to prove lower bounds on the size of proofs if we could prove lower bounds on circuits computing some particular functions. But since we are not able to prove essentially any lower bounds on general Boolean circuits, feasible interpolation gives us only conditional lower bounds. For instance, the assumption that $\mathbf{P} \neq \mathbf{NP} \cap \mathbf{coNP}$, an apparently weaker assumption than $\mathbf{NP} \neq \mathbf{coNP}$, implies that certain tautologies require superpolynomial-size proofs on systems that admit feasible interpolation.

However, in some cases, one can show that there exist monotone interpolating circuits of polynomial size provided that all variables p appear positively in $\Phi_1(p, q)$, (or negatively in $\Phi_2(p, r)$). In the case of resolution proofs, such circuits are simply monotone Boolean circuits [18, 19]. In the case of cutting-planes proofs, the interpolants are *monotone real circuits* [22]. Monotone real circuits are circuits with Boolean inputs and outputs, but whose gates are allowed to be arbitrary 2-input functions over the reals. Razborov's lower bound on the clique function has been generalized to monotone real circuits [22, 15]. Another proof system for which one can prove superpolynomial lower bounds using monotone feasible interpolation is the Nullstellensatz Proof System [23]. In this proof system, the monotone interpolants are given in terms of monotone span programs[2] [23].

[2] In the context of polynomial calculus, alternative methods (e.g. [1, 16]) yield stronger lower bounds than the monotone interpolation technique.

The results mentioned above suggest that if a proof system has the feasible interpolation property, then it may also have monotone feasible interpolation property for a suitable kind of monotone computation. The next theorem states that LS-proofs can be interpolated using MLP circuits constituted of MAX MLP gates.

▶ **Theorem 9.** *Let $\Phi(p,q) \cup \Gamma(p,r)$ be an unsatisfiable set of inequalities such that the variables $p = (p_1, ..., p_n)$ occur in Φ only with negative coefficients. Let Π be an LS refutation of $\Phi(p,q) \cup \Gamma(p,r)$. Then one can construct an MLP circuit C containing only* MAX *MLP gates which represents a Boolean function $F : \{0,1\}^n \to \{0,1\}$ such that for each $a \in \{0,1\}^n$:*
1. *if $F(a) = 1$, then $\Phi(a,q)$ is unsatisfiable,*
2. *if $F(a) = 0$, then $\Gamma(a,r)$ is unsatisfiable,*
and the size of the circuit C is polynomial in the size of Π.

4.3 Lovász-Schrijver Refutations of Mixed LP Problems

While proof systems for integer linear programming have been widely studied, very little is known about proof systems for mixed linear programming. In mixed linear programming part of variables range over integers and part of them range over reals. The Lovász-Schrijver system can naturally be adapted for mixed linear programming by disallowing the use of axioms and of the multiplication rule for variables ranging over reals. One can easily prove that the such a system is a complete refutation system (i.e., a family of inequalities is unsatisfiable iff a contradiction is derivable).

We will prove a monotone interpolation theorem for systems of mixed linear inequalities (inequalities with both types of variables) of a particular form. The advantage of this theorem, compared with the previous one, is that is uses a *single* MAX-LEFT MLP *gate* (or, by linear-programming duality, a *single* MIN-RIGHT MLP *gate*). While proving lower bounds on the size of general MLP circuits may be beyond the reach of current methods, proving a lower bound on the size of single weak MLP gate seems to be feasible, because this problem is closely related to lower bounds on extended formulations.

▶ **Theorem 10.** *Let $\Phi(p,q) \cup \Gamma(p,r)$ be a set of inequalities where p, q range over 0s and 1s, r range over reals, and the common variables $p = (p_1, ..., p_n)$ occur in Φ only with negative coefficients. Let Π be an LS-refutation of $\Phi(p,q) \cup \Gamma(p,r)$. Then there exists a* MAX-LEFT *MLP gate ℓ that represents a Boolean function $F : \{0,1\}^n \to \{0,1\}$ such that for every $a \in \{0,1\}^n$:*
1. *if $F(a) = 1$, then $\Phi(a,q)$ is unsatisfiable, and*
2. *if $F(a) = 0$, then $\Gamma(a,r)$ is unsatisfiable,*
and the size of the MLP gate ℓ is polynomial in the size of Π.

In the next subsection we will give a natural example of a set of inequalities of the form used in the theorem. We will show that it has polynomial-size mixed LS-refutations, but it requires superpolynomial-size cutting-plane refutations.

4.4 Cutting-Planes vs. Lovász-Schrijver Refutations and Monotone Real Circuits vs MLP Gates

In this subsection we will define a family Ψ_n of unsatisfiable sets of inequalities that admit polynomial-size mixed LS-refutations, but which require superpolynomial refutations in the cutting-planes proof system.

▶ **Definition 11** (Monotone Real Circuit). A monotone real circuit is a circuit C whose gates are monotone real functions of at most two variables. The size of the circuit is the number of the gates.

The following theorem can be used to translate superpolynomial lower bounds on the size of monotone real circuits computing certain partial Boolean functions into superpolynomial lower bounds for the size of cutting plane proofs.

▶ **Theorem 12** (Monotone Interpolation for the cutting-planes Proof System [22]). *Let $\Psi(p,q,r) \equiv \Phi(p,q) \cup \Gamma(p,r)$ be an unsatisfiable set of inequalities where the common variables $p = (p_1,...,p_n)$ occur in Φ only with negative coefficients. Let Π be a cutting-planes refutation for Ψ. Then one can construct a monotone real circuit C such that for every $a \in \{0,1\}^n$:*
1. *if $C(a) = 1$ then $\Phi(p,q)$ is unsatisfiable,*
2. *if $C(a) = 0$ then $\Gamma(p,q)$ is unsatisfiable,*

and the size of the circuit is at most constant time the size of the proof.

Let $K_n = \{\{i,j\} \mid 1 \leq i < j \leq n\}$ be the complete undirected graph with vertex set $[n] = \{1,...,n\}$. We say that a subgraph $X \subseteq K_n$ is a perfect matching if the edges in X are vertex-disjoint and each vertex $i \in [n]$ belongs to some edge of X. We say that a subgraph $B \subseteq K_n$ is an *unbalanced complete bipartite graph* if there exist sets $V, U \subseteq [n]$ with $V \cap U = \emptyset$, $|V| > |U|$, and $B = \{\{i,j\} \mid i \in V,\ j \in U\}$. Let $W \subseteq K_n$ be a graph. We let $\mathcal{V}(W) = \{i \mid \exists j \in [n],\ \{i,j\} \in W\}$ be the vertex set of W. For each vertex $i \in \mathcal{V}(W)$, we let $\mathcal{N}(i) = \{j \mid \{i,j\} \in W\}$ be the set of neighbors of i in W. For a subset $V \subseteq \mathcal{V}(W)$, we let $\mathcal{N}(V) = \bigcup_{v \in V} \mathcal{N}(v)$ be the set of neighbors of vertices in $\mathcal{N}(V)$. We say that W is *unbalanced* if there exists $V, U \subseteq \mathcal{V}(W)$ such that $\mathcal{N}(V) \subseteq U$ and $|V| > |U|$. Note that such an unbalanced graph W cannot contain a perfect matching X, since the existence of such a perfect matching would imply the existence of an injective mapping from V to U. We also note that unbalanced complete bipartite graphs are by definition a special case of unbalanced graphs.

Razborov showed that any monotone Boolean circuit which decides whether a graph has a perfect matching must have size at least $n^{\Omega(\log n)}$ [25]. This lower bound was generalized by Fu to the context of monotone real circuits [10]. More precisely, Fu proved that any monotone real circuit distinguishing graphs with a perfect matching from unbalanced complete bipartite graphs must have size at least $n^{\Omega(\log n)}$.

▶ **Theorem 13** ([10]). *Let $F : \{0,1\}^{\binom{n}{2}} \to \{0,1,*\}$ be a partial Boolean function such that for each $w \in \{0,1\}^{\binom{n}{2}}$:*
- *$F(w) = 1$ if w encodes a graph with a perfect matching,*
- *$F(w) = 0$ if w encodes an unbalanced complete bipartite graph.*

Then any monotone real circuit computing F must have size at least $n^{\Omega(\log n)}$.

Since unbalanced complete bipartite graphs are a special case of unbalanced graphs, monotone real circuits distinguishing graphs with a perfect matching from unbalanced graphs must have size at least $n^{\Omega(\log n)}$ gates.

▶ **Corollary 14.** *Let $g : \{0,1\}^{\binom{n}{2}} \to \{0,1,*\}$ be a partial Boolean function such that for each $w \in \{0,1\}^{\binom{n}{2}}$:*
- *$g(w) = 1$ if w has a perfect matching.*
- *$g(w) = 0$ if w is unbalanced.*

Then any monotone real circuit computing g must have size at least $n^{\Omega(\log n)}$.

Below we will define a set Ψ_n of unsatisfiable inequalities on variables:

$$p = \{w_{i,j} \mid 1 \leq i < j \leq n\},$$
$$q = \{u_i, v_i \mid i \in [n]\},$$
$$r = \{x_{ij} \mid 1 \leq i < j \leq n\}.$$

Intuitively each assignment of the variables in p defines a graph $W \subseteq K_n$ such that $\{i,j\} \in W$ if and only if $w_{ij} = 1$. Each assignment to the variables in q defines subsets $U, V \subseteq [n]$ where $i \in U$ if and only if $u_i = 1$, and $i \in V$ if and only if $v_i = 1$. Finally, each assignment to the variables in r defines a subset of edges X in such a way that $\{i,j\} \in X$ if and only if $x_{ij} = 1$. The set of inequalities Ψ_n would be satisfiable by an assignment α of the variables in p,q and r only if α defined a graph $W \subseteq K_n$ which contained, at the same time, a perfect matching X and a pair of subsets of vertices $V, U \subseteq \mathcal{V}(W)$ certifying that W is unbalanced. Since no such graph exists, the set Ψ_n is unsatisfiable.

▶ **Definition 15** (Unbalanced Graphs vs Perfect Matching Inequalities)**.** Let $\Psi_n(p,q,r) = \Phi_n(p,q) \cup \Gamma_n(p,r)$ be a set of inequalities on variables $p = \{w_{ij}\}$, $q = \{u_i, v_i\}$ and $r = \{x_{ij}\}$ defined as follows.

Inequalities in $\Phi(p,q)$:	W is unbalanced.
1) $u_j - v_i - w_{ij} + 1 \geq 0$	$\mathcal{N}(V) \subseteq U$. If $i \in V \wedge \{i,j\} \in W \Rightarrow j \in U$.
2) $\sum_j v_j - \sum_i u_i - 1 \geq 0$	$\lvert V\rvert > \lvert U\rvert$.
Inequalities in $\Gamma(p,r)$:	Existence of a perfect matching.
3) $w_{ij} - x_{ij} \geq 0$	X is a subset of edges of W.
4) $\sum_{i,i\neq j} x_{ij} - 1 = 0$	X defines a perfect matching.

Note that we can interpret the system in two alternative ways. First, all variables range over 0s and 1s. Second, variables p, q range over 0s and 1s, while r range over reals. In the second case the meaning of $\Gamma(p,q)$ is that X defines a *fractional* perfect matching. The system is, clearly, unsatisfiable also in the second case.

Note also that the variables in $w_{ij} \in p$, which occur both in $\Phi_n(p,q)$ and in $\Gamma_n(p,r)$, only occur negatively in $\Phi_n(p,q)$. A combination of Fu's size lower-bound for monotone real circuits (Theorem 13) with the monotone interpolation theorem for cutting-planes (Theorem 12) was used in [10] to show that a suitable unsatisfiable set of inequalities Ψ'_n requires cutting-planes refutations of size $n^{\Omega(\log n)}$. The next theorem states that a similar lower bound can be proved with respect to the inequalities introduced in Definition 15.

▶ **Theorem 16.** *Let $\Psi(p,q,r)$ be the set of inequalities of Definition 15 . Then any cutting-planes refutation of $\Psi(p,q,r)$ must have size at least $n^{\Omega(\log n)}$.*

On the other hand, the following theorem states that the set inequalities $\Psi_n(p,q,r)$ has LS-refutations of size polynomial in n. In fact, these refutations are for the case where variables r are real (which means that axioms and the multiplication rule is not used for variables r).

▶ **Theorem 17.** *Let $\Psi_n(p,q,r) = \Phi_n(p,q) \cup \Gamma_n(p,r)$ be the set of inequalities of Definition 15. Then $\Psi_n(p,q,r)$ has a mixed LS-refutation of size polynomial in n where r are the real variables.*

By combining Theorem 16 with Theorem 17 we have the following corollary separating cutting-planes from LS proof systems.

▶ **Corollary 18.** *The cutting-planes proof system does not polynomially simulate the Lovász-Schrijver proof system.*

Previous to our work, the problem of determining whether the cutting-planes proof system can polynomially simulate the LS-proof system had been open for almost two decades. We note that to the best of our knowledge, the converse problem, of determining whether the LS-proof system can polynomially simulate the cutting-planes proof system remains open.

By combining Theorem 17 with Theorem 10, we have that MAX-LEFT MLP gates can separate graphs with a perfect matching from unbalanced graphs superpolynomially faster than monotone real circuits. In other words, monotone real circuits cannot polynomially simulate MAX-LEFT MLP gates. We leave open the question of whether MLP gates (of any type) can polynomially simulate monotone real circuits.

▶ **Theorem 19.** *Let $g_n : \{0,1\}^{\binom{n}{2}} \to \{0,1,*\}$ be the partial Boolean function of Corollary 14. Then g_n can be represented by a* single MAX-LEFT *MLP gate of size polynomial in n.*

5 Monotone Linear Programs and Extended Formulations

A *polytope* is the convex hull of a nonempty finite set of vectors in $\mathbb{R}^n$; in particular, a polytope is *nonempty and bounded*. If a polytope $P \subseteq \mathbb{R}^n$ is given by a polynomial number of inequalities[3], then we can easily decide whether a vector $v \in \mathbb{R}^n$ belongs to P. An important observation is that even if P requires an exponential number of inequalities to be defined, we may still be able to test whether $v \in P$ efficiently if we can find a polytope $R \subseteq \mathbb{R}^{n+m}$ in a higher dimension with $m = n^{O(1)}$ such that P is a projection of P' and P' can be described by a polynomial number of inequalities[3].

More precisely, let $P \subseteq \mathbb{R}^n$ be a polytope, and let $P' \subseteq R^{n+m}$ be a polytope defined via a system of inequalities[4] $A(v,y) \leq b$. Then we say that the system $A(v,y) \leq b$ is an extended formulation of P if for each $v \in \mathbb{R}^n$, $v \in P \Leftrightarrow \exists y \in \mathbb{R}^m, A(v,y) \leq b$. We define the size of such extended formulation as the number of rows plus the number of columns in A. For instance, it can be shown that the permutahedron polytope $P_n \subseteq \mathbb{R}^n$, which is defined as the convex-hull of all permutations of the set $[n] = \{1, ..., n\}$, requires exponentially many inequalities to be defined. Nevertheless, P_n has extended formulations of size $O(n \log n)$ [12]. On the other hand, it has been shown that for some polytopes, such as the cut polytope, the TSP polytope, etc., even extended formulations require exponentially many inequalities [9, 28].

The process of defining partial Boolean functions via linear programs is closely related, but not equivalent, to the process of defining polytopes via extended formulations. For a partial Boolean function F, let $Ones(F)$, and $Zeros(F)$ denote the set of all inputs $a \in \{0,1\}^n$ such that $F(a) = 1$, and $F(a) = 0$ respectively. Let P_F^1 denote the convex hull of $Ones(F)$ and P_F^0 denote the convex hull of $Zeros(F)$. Defining F via a linear program is equivalent to finding an extended formulation of some polyhedron Q^1 that contains P_F^1 and is disjoint

[3] With coefficients specified by $n^{O(1)}$ bits.
[4] For column vectors $v \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$, (v,y) denotes the column vector $(v_1, ..., v_n, y_1, ..., y_m)$.

from $Zeros(F)$, or an extended formulation of some polyhedron Q^0 that contains P^0_F and is disjoint from $Ones(F)$. Finding such an extended formulation for Q^1 (resp. Q^0) with a small number of inequalities is clearly, a simpler task than finding a small extended formulation for the polyhedron P^1_F (resp. P^0_F) itself. For instance, if F is the matching function for general graphs, then F is computable by a polynomial-size Boolean circuit (containing negation gates), and hence this function can be defined via (not necessarily monotone) linear programs of polynomial size[5]. Nevertheless, the corresponding polytope P^1_F requires extended formulations of exponential size [28].

Let $F : \{0,1\}^n \to \{0,1,*\}$ be a partial monotone Boolean function. A minterm of F is a vector $v \in \{0,1\}^n$ such that $F(v) = 1$ and such that $F(v') \neq 1$ for each $v' \leq v$. Intuitively, a minterm is a minimal vector which causes F to evaluate to 1. Analogously, a maxterm is a vector $v \in \{0,1\}^n$ such that $F(v) = 0$ and $F(v') \neq 0$ for each $v \geq 0$. Intuitively, a maxterm is a maximal vector that causes F to evaluate to 0. We let $\hat{P}^1_F$ be the convex-hull of minterms of F, and let $\hat{P}^0_F$ be the convex-hull of maxterms of F. Let H^1 be a hyperplane containing $\hat{P}^1_F$. For each maxterm v we define the set $S^1_v = H^1 \cap \{u \mid u \leq v\}$. Analogously, let H^0 be an hyperplane containing $\hat{P}^0_F$. We define the set $S^0_v = H^0 \cap \{u \mid u \geq v\}$.

▶ **Definition 20** (Monotone Extension Complexity)**.** Let $F : \{0,1\}^n \to \{0,1,*\}$ be a partial monotone Boolean function. Below we define two notions of monotone extension complexity (mxc) for F.

1. We let $mxc_1(F)$ denote the minimum size of an extended formulation for a polytope Q^1 such that

$$\hat{P}^1_F \subseteq Q^1, \qquad \text{and} \qquad Q \cap \bigcup_v S^1_v = \emptyset. \tag{2}$$

2. We let $mxc_0(F)$ denote the minimum size of an extended formulation for a polytope Q^0 such that

$$\hat{P}^0_F \subseteq Q^0, \qquad \text{and} \qquad Q \cap \bigcup_v S^0_v = \emptyset. \tag{3}$$

The next theorem relates the monotone extension complexity of a partial monotone Boolean function F to the size of MLP representations for F.

▶ **Theorem 21.** *Let $F : \{0,1\}^n \to \{0,1,*\}$ be a partial monotone Boolean function. Then $mxc_1(F)$ is up to a constant factor equal to the minimum size of a* MAX-RIGHT *MLP representation of F. Analogously, the $mxc_0(F)$ is up to a constant factor equal to the minimum size of a* MIN-LEFT MLP *representation of F.*

6 Conclusion

In this work we introduced several models of computation based on the notion of monotone linear programs. In particular, we introduced the notions of weak and strong MLP gates. We reduced the problem of proving lower bounds for the size of LS proofs to the problem of proving lower bounds for the size of MLP circuits with strong gates, and the problem of proving lower bounds on the size of mixed LS proofs to the problem of proving lower bounds on the size of single weak MLP gates.

[5] Note that any function in PTIME can be defined by polynomial-size non-monotone LP programs, due to the fact that linear programming is PTIME complete.

When it comes to comparing MLP gates with other models of computation, we have shown that weak MLP gates are strictly more powerful than monotone Boolean circuits and monotone span programs. Additionally, these gates cannot be polynomially simulated by monotone real circuits. Finally, by combining some results mentioned above, we proved that the cutting-planes proof system is not powerful enough to polynomially simulate the LS proof system. This is the first result showing a separation between the power of these two systems.

Acknowledgments. We would like to thank Pavel Hrubeš and Massimo Lauria for discussion and their valuable suggestions.

References

1 Michael Alekhnovich and Alexander A. Razborov. Satisfiability, branch-width and Tseitin tautologies. In *Proc. of the 43rd Symposium on Foundations of Computer Science*, pages 593–603, 2002.

2 László Babai, Anna Gál, and Avi Wigderson. Superpolynomial lower bounds for monotone span programs. *Combinatorica*, 19(3):301–319, 1999.

3 Paul Beame, Russell Impagliazzo, Jan Krajíček, Toniann Pitassi, and Pavel Pudlák. Lower bounds on Hilbert's Nullstellensatz and propositional proofs. *Proceedings of the London Mathematical Society*, 3(1):1–26, 1996.

4 Gábor Braun, Samuel Fiorini, Sebastian Pokutta, and David Steurer. Approximation limits of linear programs (beyond hierarchies). *Mathematics of Operations Research*, 40(3):756–772, 2015.

5 Mark Braverman and Ankur Moitra. An information complexity approach to extended formulations. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing*, pages 161–170. ACM, 2013.

6 Samuel R. Buss and Toniann Pitassi. Good degree bounds on Nullstellensatz refutations of the induction principle. *Journal of Computer and System Sciences*, 57(2):162–171, 1998.

7 Stephen A. Cook, Toniann Pitassi, Robert Robere, and Benjamin Rossman. Exponential lower bounds for monotone span programs. *ECCC*, TR16-64, 2016.

8 Sanjeeb Dash. Exponential lower bounds on the lengths of some classes of branch-and-cut proofs. *Mathematics of Operations Research*, 30(3):678–700, 2005.

9 Samuel Fiorini, Serge Massar, Sebastian Pokutta, Hans Raj Tiwary, and Ronald De Wolf. Exponential lower bounds for polytopes in combinatorial optimization. *Journal of the ACM (JACM)*, 62(2):17, 2015.

10 Xudong Fu. Lower bounds on sizes of cutting planes proofs for modular coloring principles. *Proof Complexity and Feasible Arithmetics*, pages 135–148, 1998.

11 Anna Gál and Pavel Pudlák. A note on monotone complexity and the rank of matrices. *Information Processing Letters*, 87(6):321–326, 2003.

12 Michel X. Goemans. Smallest compact formulation for the permutahedron. *Mathematical Programming*, 153(1):5–11, 2015.

13 Dima Grigoriev. Linear lower bound on degrees of Positivstellensatz calculus proofs for the parity. *Theoretical Computer Science*, 259(1):613–622, 2001.

14 Armin Haken. The intractability of resolution. *Theoretical Computer Science*, 39:297–308, 1985.

15 Armin Haken and Stephen A. Cook. An exponential lower bound for the size of monotone real circuits. *Journal of Computer and System Sciences*, 58(2):326–335, 1999.

16 Russell Impagliazzo, Pavel Pudlák, and Jiri Sgall. Lower bounds for the polynomial calculus and the gröbner basis algorithm. *Computational Complexity*, 8(2):127–144, 1999.

17 M. Karchmer and A. Wigderson. On span programs. In *Proceedings of the 8th Annual Structure in Complexity Theory Conference*, pages 102–111. IEEE CS, 1993.

18 Jan Krajíček. Interpolation theorems, lower bounds for proof systems, and independence results for bounded arithmetic. *The Journal of Symbolic Logic*, 62(02):457–486, 1997.

19 Jan Krajíček. Interpolation and approximate semantic derivations. *Mathematical Logic Quarterly*, 48(4):602–606, 2002.

20 László Lovász and Alexander Schrijver. Cones of matrices and set-functions and 0-1 optimization. *SIAM Journal on Optimization*, 1(2):166–190, 1991.

21 Toniann Pitassi and Nathan Segerlind. Exponential lower bounds and integrality gaps for tree-like lovasz-schrijver procedures. *SIAM Journal on Computing*, 41(1):128–159, 2012.

22 Pavel Pudlák. Lower bounds for resolution and cutting plane proofs and monotone computations. *The Journal of Symbolic Logic*, 62(03):981–998, 1997.

23 Pavel Pudlak and Jiri Sgall. Algebraic models of computation and interpolation for algebraic proof systems. In *Proc. of Feasible Arithmetic and Proof Complexity, DIMACS Series in Discrete Math. and Theoretical Comp. Sci.*, volume 39, pages 279–295, 1998.

24 Ran Raz and Avi Wigderson. Monotone circuits for matching require linear depth. *Journal of the ACM (JACM)*, 39(3):736–744, 1992.

25 Alexander A. Razborov. Lower bounds on monotone complexity of the logical permanent. *Mathematical Notes*, 37(6):485–493, 1985.

26 Alexander A. Razborov. Lower bounds for monotone complexity of boolean functions. *American Mathematical Society Translations*, 147:75–84, 1990.

27 Alexander A. Razborov. Proof complexity and beyond. *ACM SIGACT News*, 47(2):66–86, 2016.

28 Thomas Rothvoß. The matching polytope has exponential extension complexity. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, pages 263–272. ACM, 2014.

29 Alexander Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency*, volume 24 of *Algorithms and Combinatorics*. Springer, 2003.

A Note on Amortized Branching Program Complexity*

Aaron Potechin

Institute for Advanced Study, Princeton, NJ, USA
aaronpotechin@gmail.com

Abstract

In this paper, we show that while almost all functions require exponential size branching programs to compute, for all functions f there is a branching program computing a doubly exponential number of copies of f which has linear size per copy of f. This result disproves a conjecture about non-uniform catalytic computation, rules out a certain type of bottleneck argument for proving non-monotone space lower bounds, and can be thought of as a constructive analogue of Razborov's result that submodular complexity measures have maximum value $O(n)$.

1998 ACM Subject Classification F.1.1 Models of Computation

Keywords and phrases branching programs, space complexity, amortization

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.4

1 Introduction

In amortized analysis, which appears throughout complexity theory and algorithm design, rather than considering the worst case cost of an operation, we consider the average cost of the operation when it is repeated many times. This is very useful in the situation where operations may have a high cost but if so, this reduces the cost of future operations. In this case, the worst-case rarely occurs and the average cost of the operation is much lower. A natural question we can ask is as follows. Does amortization only help for specific operations, or can any operation/function be amortized?

For boolean circuits, which are closely related to time complexity, Uhlig [12],[13] showed that for any function f, as long as m is $2^{o(\frac{n}{\log n})}$ there is a circuit of size $O(\frac{2^n}{n})$ computing f on m different inputs simultaneously. As shown by Shannon [11] and Lupanov [6], almost all functions require circuits of size $\Theta(\frac{2^n}{n})$ to compute, which means that for almost all functions f, the cost to compute many inputs of f is essentially the same as the cost to compute one input of f!

In this paper, we consider a similar question for branching programs, which are closely related to space complexity. In particular, what is the minimum size of a branching program which computes many copies of a function f on the same input? This question is highly non-trivial because branching programs are not allowed to copy bits, so we cannot just compute f once and then copy it. In this paper, we show that for $m = 2^{2^n-1}$, there is a branching program computing m copies of f which has size $O(mn)$ and thus has size $O(n)$ per copy of f.

This work has connections to several other results in complexity theory. In catalytic computation, introduced by Buhrman, Cleve, Koucký, Loff, and Speelman [2], we have an

* This material is based upon work supported by the National Science Foundation under agreement No. CCF-1412958 and by the Simons Foundation.

32nd Computational Complexity Conference (CCC 2017).
Editor: Ryan O'Donnell; Article No. 4; pp. 4:1–4:12

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

additional tape of memory which is initially full of unknown contents. We are allowed to use this tape, but we must restore it to its original state at the end of our computation. As observed by Girard, Koucký, and McKenzie [5], the model of a branching program computing multiple instances of a function is a non-uniform analogue of catalytic computation and our result disproves Conjecture 25 of their paper. Our result also rules out certain approaches for proving general space lower bounds. In particular, any lower bound technique which would prove a lower bound on amortized branching program complexity as well as branching program size cannot prove non-trivial lower bounds. Finally, our result is closely related to Razborov's result [10] that submodular complexity measures have maximum size $O(n)$ and can be thought of as a constructive analogue of Razborov's argument.

1.1 Outline

In Section 2 we give some preliminary definitions. In section 3 we give our branching program construction, proving our main result. In section 4 we briefly describe the relationship between our work and catalytic computation. In section 5 we discuss which lower bound techniques for proving general space lower bounds are ruled out by our construction. Finally, in section 6 we describe how our work relates to Razborov's result [10] on submodular complexity measures.

2 Preliminaries

In this section, we define branching programs, branching programs computing multiple copies of a function, and the amortized branching program complexity of a function.

▶ **Definition 1.** We define a branching program to be a directed acyclic multi-graph B with labeled edges and distinguished start nodes, accept nodes, and reject nodes which satisfies the following conditions.

1. Every vertex of B has outdegree 0 or 2. For each vertex $v \in V(B)$ with outdegree 2, there exists an $i \in [1, n]$ such that one of the edges going out from v has label $x_i = 0$ and the other edge going out from v has label $x_i = 1$.
2. Every vertex with outdegree 0 is an accept node or a reject node.

Given a start node s of a branching program and an input $x \in \{0,1\}^n$, we start at s and do the following at each vertex v that we reach. If v is an accept or reject node then we accept or reject, respectively. Otherwise, for some i, one of the labels going out from v has label $x_i = 0$ and the other edge going out from v has label $x_i = 1$. If $x_i = 0$ then we take the edge with label $x_i = 0$ and if $x_i = 1$ then we take the edge with label $x_i = 1$. In other words, we follow the path starting at s whose edge labels match x until we reach an accept or reject node and accept or reject accordingly.

Given a branching program B and start node s, let $f_{B,s}(x) = 1$ if we reach an accept node when we start at s on input x and let $f_{B,s} = 0$ if we reach a reject node when we start at s on input x. We say that (B, s) computes $f_{B,s}$.

We define the size of a branching program B to be $|V(B)|$, the number of vertices/nodes of B.

▶ Remark. We can consider branching programs whose start nodes all compute different functions, but for this note we will focus on branching programs whose nodes all compute the same function.

▶ **Definition 2.** We say that a branching program B computes f m times if B has m start nodes $s_1, \cdots, s_m$ and $f_{B,s_i} = f$ for all i.

▶ Remark. In the case where $m = 1$ we recover the usual definition of a branching program B computing a function f: if $f(x) = 1$ then B goes from s to an accept node on input x and if $f(x) = 0$ then B goes from s to a reject node on input x.

▶ **Definition 3.** We say that a branching program B is index-preserving if there is an m such that:

1. B has m, start nodes $s_1, \cdots, s_m$, m accept nodes $a_1, \cdots, a_m$, and m reject nodes $r_1, \cdots, r_m$,
2. for all i and all inputs x, if B starts at s_i on input x then it will either end on a_i or r_i.

▶ **Definition 4.** Given a function f.

1. We define $b_m(f)$ to be the minimal size of an index-preserving branching program which computes f m times.
2. We define the amortized branching program complexity $b_{avg}(f)$ of f to be

$$b_{avg}(f) = \lim_{m \to \infty} \frac{b_m(f)}{m}.$$

▶ **Proposition 5.** *For all functions f, $b_{avg}(f)$ is well-defined and is equal to* $\inf\{\frac{b_m(f)}{m} : m \geq 1\}$.

Proof. Note that for all $m_1, m_2 \geq 1$, $b_{m_1+m_2}(f) \leq b_{m_1}(f) + b_{m_2}(f)$ as if we are given a branching program computing f m_1 times and a branching program computing f m_2 times, we can take their disjoint union and this will be a branching program computing f $m_1 + m_2$ times. Thus for all $m_0 \geq 1$, $k \geq 1$, and $0 \leq r < m_0$, $b_{km_0+r}(f) \leq kb_{m_0}(f) + b_r(f)$. This implies that $\lim_{m\to\infty} \frac{b_m(f)}{m} \leq \frac{b_{m_0}(f)}{m_0}$ and the result follows. ◀

3 The Construction

In this section, we give our construction of a branching program computing doubly exponentially many copies of a function f which has linear size per copy of f, proving our main result.

▶ **Theorem 6.** *For all f, $b_{avg}(f) \leq 64n$. In particular, for all f, taking $m = 2^{2^n-1}$, $b_m(f) \leq 32n2^{2^n}$.*

Proof. Our branching program has several parts. We first describe each of these parts and how we put them together and then we will describe how to construct each part. The first two parts are as follows:

1. A branching program which simultaneously identifies all functions $g : \{0,1\}^n \to \{0,1\}$ that have value 1 for a given x. More preceisely, it has start nodes $s_1, \cdots, s_m$ where $m = 2^{2^n-1}$ and has one end node t_g for each possible function $g : \{0,1\}^n \to \{0,1\}$, with the guarantee that if $g(x) = 1$ for a given g and x then there exists an i such that that the branching program goes from s_i to t_g on input x.
2. A branching program which simultaneously evaluates all functions $g : \{0,1\}^n \to \{0,1\}$. More precisely, it has one start node s_g for each function g and has end nodes $a'_1, \cdots, a'_m$ and $r'_1, \cdots, r'_m$, with the guarantee that for a given g and x, if $g(x) = 1$ then the branching program goes from s_g to a'_i for some i and if $g(x) = 0$ then the branching program goes from s_g to r'_i for some i.

If f is the function which we actually want to compute, we combine these two parts as follows. The first part gives us paths from $\{s_i : i \in [1, m]\}$ to $\{t_g : g(x) = 1\}$. We now take each t_g from the first part and set it equal to $s_{(f\wedge g)\vee(\neg f\wedge\neg g)}$ in the second part. Once

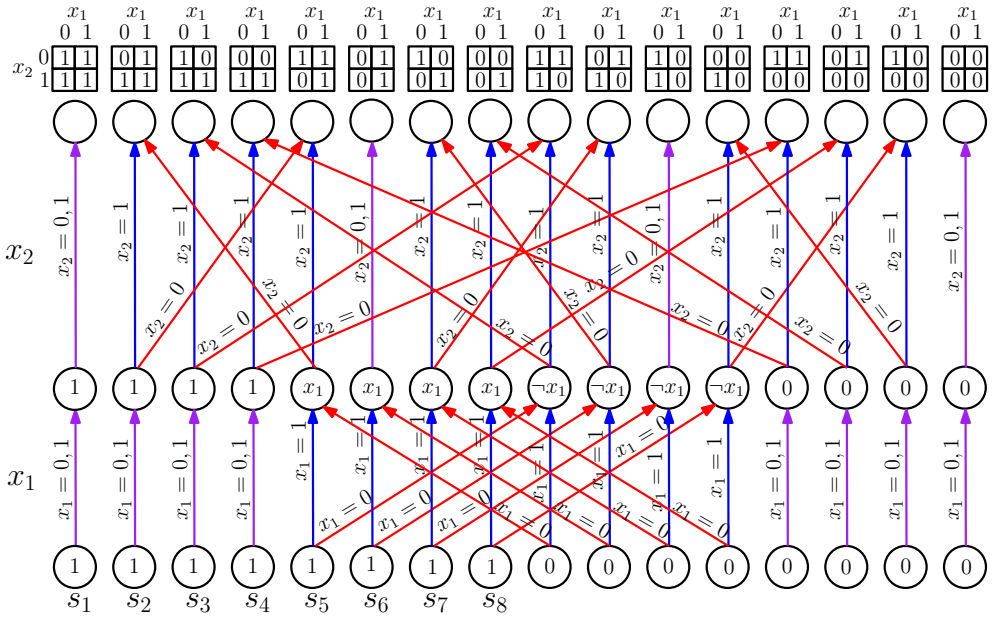


Figure 1 This figure illustrates part 1 of our construction for $n = 2$. The functions for the top vertices are given by the truth tables at the top and each other vertex corresponds to the function inside it. Blue edges can be taken when the corresponding variable has value 1, red edges can be taken when the corresponding variable has value 0, and purple edges represent both a red edge and a blue edge (which are drawn as one edge to make the diagram cleaner). Note that for all inputs x, there are paths from the start nodes to the functions which have value 1 on input x at each level.

we do this, if $f(x) = 1$ then for all g, $g(x) = 1 \iff (f \wedge g) \vee (\neg f \wedge \neg g) = 1$ so we will have paths from $\{t_g : g(x) = 1\} = \{s_{(f\wedge g)\vee(\neg f\wedge\neg g)} : g(x) = 1\}$ to $\{a_i : i \in [1, m]\}$. If $f(x) = 0$ then for all g, $g(x) = 1 \iff (f \wedge g) \vee (\neg f \wedge \neg g) = 0$, so we will have paths from $\{t_g : g(x) = 1\} = \{s_{(f\wedge g)\vee(\neg f\wedge\neg g)} : g(x) = 1\}$ to $\{r_i : i \in [1, m]\}$. Putting everything together, when $f(x) = 1$ we will have paths from $\{s_i : i \in [1, m]\}$ to $\{a_i : i \in [1, m]\}$ and when $f(x) = 0$ we will have paths from $\{s_i : i \in [1, m]\}$ to $\{r_i : i \in [1, m]\}$.

Combining these two parts gives us a branching program B which computes f m times. However, these paths do not have to map s_i to a'_i or r'_i, they can permute the final destinations. In other words, B will not be index-preserving. To fix this, we construct a final part which runs the branching program we have so far in reverse. If we added this final part to B, this would fix the permutation issue but would get us right back where we started! To avoid this, we instead have two copies of the final part, one applied to $\{a'_i : i \in [1, m]\}$ and one applied to $\{r'_i : i \in [1, m]\}$. This separates the case when $f(x) = 1$ and the case $f(x) = 0$, giving us our final branching program.

We now describe how to construct each part. For the first part, which simultaneously identifies the functions which have value 1 on input x, we have a layered branching program with $n+1$ levels going from 0 to n. At level j, for each function $g : \{0,1\}^j \to \{0,1\}$, we have $2^{2^n-2^j}$ nodes corresponding to g. For all $j \in [1, n]$ we draw the arrows from level $j-1$ to level j as follows. For a node corresponding to a function $g : \{0,1\}^{j-1} \to \{0,1\}$, we draw an arrow with label $x_j = 1$ from it to a node corresponding to a function $g' : \{0,1\}^j \to \{0,1\}$ such that $g'(x_1, \cdots, x_{j-1}, 1) = g(x_1, \cdots, x_{j-1})$. Similarly, we draw an arrow with label $x_j = 0$ from it to a node corresponding to a function $g' : \{0,1\}^j \to \{0,1\}$ such that $g'(x_1, \cdots, x_{j-1}, 0) = g(x_1, \cdots, x_{j-1})$. We make these choices arbitrarily but make sure that no two arrows with the same label have the same destination.

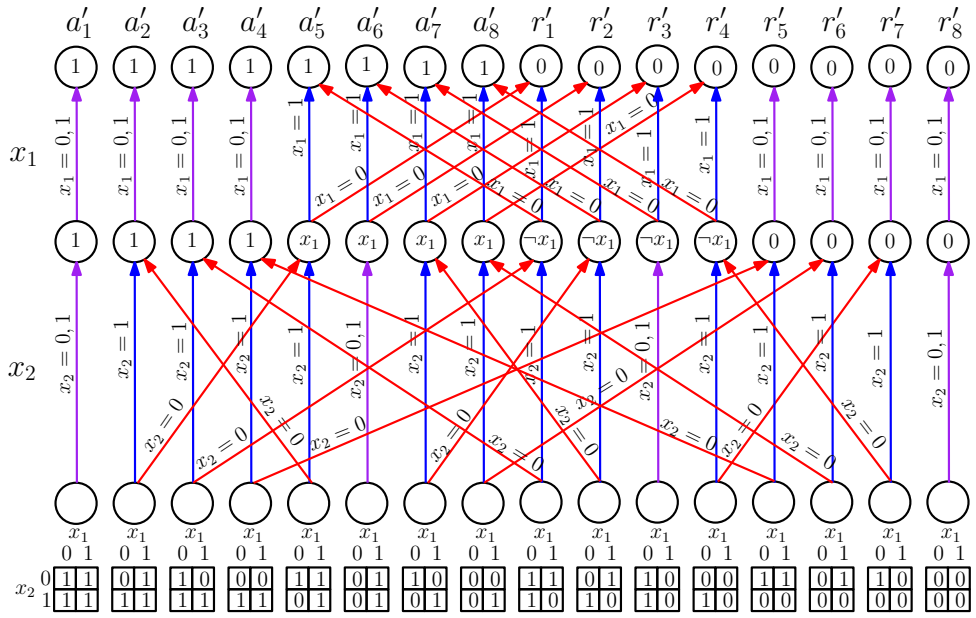


Figure 2 This figure illustrates part 2 of our construction for $n = 2$. The functions for the bottom vertices are given by the truth tables at the bottom and each other vertex corresponds to the function inside it. Note that for all inputs x, the paths go between the functions which evaluate to 1 on input x and the accept nodes and between the functions which evaluate to 0 on input x and the reject nodes.

For the second part, which simultaneously evaluates each function, we have a layered branching program with $n + 1$ levels going from 0 to n. At level $n - j$, for each function $g : \{0,1\}^j \to \{0,1\}$, we have $2^{2^n - 2^j}$ nodes corresponding to g. For all $j \in [1, n]$ we draw the arrows from level $n - j$ to level $n - j + 1$ as follows. For a node corresponding to a function $g : \{0,1\}^j \to \{0,1\}$, we draw an arrow with label $x_j = 1$ from it to a node corresponding to the function $g(x_1, \cdots, x_{j-1}, 1)$ and draw an arrow with label $x_j = 0$ from it to a node corresponding to the function $g(x_1, \cdots, x_{j-1}, 0)$. Again, we make these choices arbitrarily but make sure that no two edges with the same label have the same destination.

For the final part, note that because we made sure not to have any two edges with the same label have the same destination and each level has the same number of nodes, our construction so far must have the following properties:

1. Every vertex has indegree 0 or 2. For the vertices v with indegree 2, there is a j such that one edge going into v has label $x_j = 1$ and the other edge going into v has label $x_j = 0$.
2. The vertices which have indegree 0 are precisely the vertices in the bottom level.

These conditions imply that if we reverse the direction of each edge in the branching program we have so far, this gives us a branching program which runs our branching program in reverse. As described before, we now take two copies of this reverse program. For one copy, we take its start nodes to be $a'_1, \cdots, a'_m$ and relabel its copies of $s_1, \cdots, s_m$ as $a_1, \cdots, a_m$. For the other copy, we take its start nodes to be $r'_1, \cdots, r'_m$ and relabel its copies of $s_1, \cdots, s_m$ as $r_1, \cdots, r_m$. ◀

4 Relationship to catalytic computation

In catalytic computation, we have additional memory which we may use but this memory starts with unknown contents and we must restore this memory to its original state at the

end. Our result is related to catalytic computation through Proposition 9 of [5], which says the following

▶ **Proposition 7.** *Let f be a function which can be computed in space $s(n)$ using catalytic tape of size $l(n) \leq 2^{s(n)}$. Then $b_{2^{l(n)}}(f)$ is $2^{l(n)} \cdot 2^{O(s(n))}$.*

For convenience, we give a proof sketch of this result here.

Proof Sketch. This can be proved using the same reduction that is used to reduce a Turing machine using space $s(n)$ to a branching program of size $2^{O(s(n))}$, with the following differences. There are $2^{l(n)}$ possibilities for what is in the catalytic tape at any given time, so the resulting branching program is a factor of $2^{l(n)}$ times larger. The requirement that the catalytic tape is restored to its original state at the end implies that there must be $2^{l(n)}$ disjoint copies of the start, accept, and reject nodes, one for each possibility for what is in the catalytic tape originally. This means that the branching program computes f $2^{l(n)}$ times. Finally, the condition that $l(n) \leq 2^{s(n)}$ is necessary because otherwise the branching program would have to be larger in order to keep track of where the pointer to the catalytic tape is pointing! ◀

Girard, Koucký, and McKenzie [5] conjectured that for a random function f, for all $m \geq 1$, $b_m(f)$ is $\Omega(mb_1(f))$. If true, this conjecture would imply (aside from issues of non-uniformity) that a catalytic tape does not significantly reduce the space required for computing most functions. However, our construction disproves this conjecture.

That said, our construction requires m to be doubly exponential in n. It is quite possible that $\log(\frac{b_m(f)}{m})$ is $\Omega(\log(b_1(f)))$ for much smaller m, which would still imply (aside from issues of non-uniformity) that a catalytic tape does not reduce the space required for computing most functions by more than a constant factor.

5 Barrier for input-based bottleneck arguments

As noted in the introduction, our result rules out any general lower bound approach which would prove lower bounds on amortized branching program complexity as well as branching program size. In this section, we discuss one such class of techniques.

One way we could try to show lower bounds on branching programs is as follows. We could argue that for the given function f and a given branching program B computing f, for every YES input x the path that B takes on input x contains a vertex giving a lot of information about x and thus G must be large to accomodate all of the possible inputs. We observe that this kind of argument would show lower bounds on amortized branching program complexity as well as on branching program size and thus cannot show nontrivial general lower bounds. We assume for the remainder of the section that we are trying to compute some function $f : \{0,1\}^n \to \{0,1\}$ with a branching program of a given type.

▶ **Definition 8.** We define a function description h to be a mapping which takes a branching program B and assigns a funtion $h_v : \{0,1\}^n \to \Omega$ to every vertex v of B. Here Ω can be an arbitrary set but we will focus on the case when $\Omega = \{0,1\}$.

▶ **Example 9.** A particularly useful function description is the reachability function description which sets $h_v(x) = 1$ if it is possible to reach v from a start node on input x and sets $h_v(x) = 0$ otherwise.

▶ **Definition 10.** We define a bottleneck criterion c to be a mapping which takes a function $g : \{0,1\}^n \to \Omega$ and an $x \in \{0,1\}^n$ and outputs 0 or 1. The idea is that $c(g,x) = 1$ if g gives a lot of information about x and $c(g,x) = 0$ otherwise.

▶ **Definition 11.** We say that a function description and bottleneck criterion (h, c) are valid for a given type of branching program if for any branching program B of this type, any YES input x, and any path P in B from a start node to an accept node on input x there is a vertex $v \in V(P)$ such that $c(h_v, x) = 1$.

▶ **Definition 12.** We say that a bottleneck criterion c has selectivity S_c if there is a set of YES inputs I such that for all g, there are at most $\frac{|I|}{S_c}$ inputs $x \in I$ such that $c(g, x) = 1$

We note that bottleneck criteria with high selectivity imply large lower size bounds on the given type of branching program.

▶ **Proposition 13.** *If there exists a valid function description and bottleneck criterion (h, c) for a given type of branching program and c has selectivity S_c then any branching program of the given type computing f must have size at least S_c.*

We now observe that bottleneck criteria with high selectivity also imply large lower size bounds on amortized branching programs.

▶ **Lemma 14.** *If there exists a valid function description and bottleneck criterion (h, c) for a given type of branching program and c has selectivity S_c then any branching program of the given type computing f m times has size at least mS_c*

Proof. Let B be a branching program computing f m times. Let N be the total number of times that we have a vertex $v \in V(B)$ and an input $x \in I$ such that $c(h_v, x) = 1$. On the one hand, $N \geq m|I|$ as for each $x \in I$ there are m disjoint paths in B from a start node to an accept node, each of which must contain a vertex v such that $c(h_v, x) = 1$ (as otherwise (h, c) wouldn't be valid). On the other hand, since c has selectivity S_c, for each $v \in V(B)$ there are at most $\frac{|I|}{S_c}$ $x \in I$ such that $c(h_v, x) = 1$. Thus, $N \leq \frac{|V(B)| \cdot |I|}{S_c}$. Putting these two bounds together, we have that $\frac{|V(B)| \cdot |I|}{S_c} \geq m|I|$ which implies that $|V(B)| \geq mS_c$, as needed. ◀

▶ **Corollary 15.** *Given a valid function description and bottleneck criterion (h, c) for general branching programs, $S_c \leq 64n$*

Proof. By Lemma 14, if (h, c) is a valid function description and bottleneck criterion and c has selectivity S_c then for all m, $b_m(f) \geq mS_c$. However, Theorem 6 says that for $m = 2^{2^n - 1}$, $b_m(f) \leq 64mn$. Thus, we must have that $S_c \leq 64n$, as needed. ◀

▶ Remark. To prove such an upper bound on S_c it is sufficient to find a B which computes f m times. B does not have to be index-preserving. As noted in the proof of Theorem 6, the first two parts of the construction in Theorem 6 are sufficient to construct such a B. Thus, we have the same upper bound on S_c even for oblivious read-twice branching programs (a branching program is oblivious if it reads the input bits in the same order regardless of the input)!

5.1 Examples of input-based bottleneck arguments

In this subsection, we briefly discuss which lower bound approaches can be viewed as input-based bottleneck arguments. In particular, we note that the current framework of Potechin and Chan [3] for analyzing monotone switching networks can be seen as an input-based bottleneck argument, as can many lower bounds on read-once branching programs.

▶ **Example 16** (Fourier analysis on monotone switching networks)**.** At a high level, the current framework of Potechin and Chan [3] for analyzing monotone switching networks works as follows:

1. Take I to be a large set of minimal YES inputs which are almost disjoint from each other.
2. Use the reachability function description, focusing on the maximal NO instances (the cuts).
3. Carefully construct a set of functions g_{xi} for each $x \in I$ and use the criterion $c(h_v, x) = 1$ if $|\langle g_{xi}, h_v \rangle| \geq \frac{a}{l}$ for some i and is 0 otherwise, where $a > 0$ is a constant and l is the maximum length of an accepting path in the switching network. The intuition is that the functions g_{xi} pick out high-degree information about the input x which much be processed to accept x, so any accepting path for x must contain a vertex v where $|\langle g_{xi}, h_v \rangle|$ is large for some i.
4. Use Fourier analysis to argue that c has high selectivity.

▶ **Example 17** (k-clique)**.** Wegener [14] and Zak [15] independently proved exponential lower bounds on the size of read-once branching programs solving the problem of whether a graph G has a clique of linear size. To prove their lower bounds, they argue that near the start node, the branching program must branch off like a tree or else it cannot be completely accurate. This kind of argument is not captured with an input-based bottleneck argument, as it uses the structure of the given braching program. That said, we can prove a $\frac{\binom{n}{k}}{n-k+1}$ lower size bound on read-once branching programs for k-clique with the following input-based bottleneck argument:

1. We take c to be the following bottleneck criterion. We take I to be the set of minimal YES instances, i.e. graphs which have a clique of size k and no other edges. Since we are considering a read-once branching program, for each node v we have a partition of the input bits based on whether they are examined before or after reaching v (input bits which are never examined on any computation path containing v can be put in either side). Given an $x \in I$, this induces a partition (E_1, E_2) of the edges of the k-clique in x. We take $c(v, x) = 1$ if there is a path from a start node to an accept node on input x which passes through v and we have that E_1 contains $k-2$ of the edges incident to some vertex u in the k-clique but there is no vertex u in the k-clique such that E_1 contains all $k-1$ edges incident with u.
2. We argue that c has high selectivity as follows. If $c(v, x) = c(v, x') = 1$ for some v, x, x' then consider the corresponding partitions (E_1, E_2) and (E_1', E_2'). $E_1 \cup E_2'$ and $E_1' \cup E_2$ must form k-cliques so we must have that $|E_1| = |E_1'|$ and $|E_2| = |E_2'|$ and $E_1 \cup E_2', E_1' \cup E_2$ contain no extra edges.
   Now let A be the set of vertices w of the k-clique in x such that both E_1 and E_2 contain some but not all of the edges incident to w in x. We observe that the k-clique in x' contains A as otherwise the edges in $E_1 \cup E_2'$ and $E_1' \cup E_2$ incident to w are wasted which is impossible as $E_1 \cup E_2'$ and $E_1' \cup E_2$ can have no extra edges. Thus, we can only have $c(v, x') = 1$ for the x' such that the k-clique contains A.
   From the definition of c, A must include some vertex u in the k-clique of x and $k-2$ of its neighbors, so $|A| \geq k-1$. This implies that $c(v, x') = 1$ for at most $n-k+1$ distinct x'. The total number of $x' \in I$ is $\binom{n}{k}$ so our lower bound is $\frac{\binom{n}{k}}{n-k+1}$.

▶ **Example 18** (Majority)**.** In his master's thesis introducing branching programs, Masek [7] proved a quadratic lower bound on the size of read-once branching programs computing the majority function. With an input-based bottleneck argument, we obtain a lower bound of $\Omega(n^{\frac{3}{2}})$, which is lower, but there is a reason for this. We can prove our lower bound as follows. Here we assume that $n \geq 3$ is odd.

1. We take I to be the set of inputs with exactly $\frac{n+1}{2}$ ones.
2. We note that in order for the branching program to be read-once and be correct on all inputs from all starting nodes, we must have that for each node v of the branching program, there is a partition (A, B) of the inputs bits such that on all paths from a start node to v, only input bits in A are examined and on all paths from v to an accept node or reject node, only inputs in B are examined. Moreover, if $|A| < \frac{n}{2}$ then every path from a start node to v must examine all of the bits in A and must have the same number of these bits equal to one.
3. We choose an $m < \frac{n}{2}$ and take $c(v, x) = 1$ if $|A| = m$ and there is a path from a start node to v on input x and we take $c(v, x) = 0$ otherwise. Note that for any vertex v, all of the x such that $c(v, x) = 1$ have the same number of ones in A so there can be at most $O(\frac{|I|}{\sqrt{m}})$ such x and c has selectivity $\Omega(\sqrt{m})$.
4. We sum this over all $m < \frac{n}{2}$ obtaining our final lower bound of $\Omega(n^{\frac{3}{2}})$

▶ Remark. With a more complicated argument, it can be shown that this lower bound applies even if we allow the branching program to reject a small portion of the YES inputs (while still requiring that it rejects all NO inputs). This is a reason why we only obtain a lower bound of $\Omega(n^{\frac{3}{2}})$ rather than $\Omega(n^2)$; we can probabilistically choose a branching program of size $O(n^{\frac{3}{2}}log(n))$ for majority which rejects all NO inputs and accepts any given YES input with very high probability.

▶ Remark. If we assume that our branching program is oblivious as well as read-once (in which case we can assume without loss of generality that the input bits are read in order) then we can prove an $\Omega(n^2)$ lower bound using an input-based bottleneck argument. The idea is that we can take a different set of inputs I in order to increase the selectivity of our bottleneck criterion c. In particular, for each m we can take a set of inputs I_m such that I_m contains $m + 1$ minimal YES inputs, each with a different number of ones in the first m bits. This c now has selectivity m and summing over all $m < \frac{n}{2}$ gives a lower bound of $\Omega(n^2)$

As these examples show, input-based bottleneck arguments are effective for proving lower bounds on read-once branching programs. Thus, Theorem 6 and Lemma 14, which together rule out input-based bottleneck arguments even for oblivious read-twice branching programs, provide considerable insight into the spike in difficulty between proving lower bounds for read-once branching programs and read-twice branching programs which can be seen in Razborov's survey [9] on branching programs and related models. That said, Theorem 6 and Lemma 14 do not say anything about lower bounds based on counting functions such as Nechiporuk's quadratic lower bound [8] or lower bounds based on communication complexity arguments such as Babai, Nisan, and Szegedy's result [1] proving an exponential lower bound on oblivious read-k branching programs for arbitrary k. We also note that Cook, Edmonds, Medabalimi, and Pitassi [4] give another explanation for the failure of bottleneck arguments past read-once branching programs.

6 Linear upper bound on complexity measures

Another way we could try to lower bound branching program size is through a complexity measure on functions. However, Razborov [10] showed that submodular complexity measures cannot have superlinear values. In this section we show that this is also true for a similar class of complexity measures, branching complexity measures, which correspond more closely to branching programs. We then show that all submodular complexity measures are also branching complexity measures, so Theorem 6 is a constructive analogue and a slight generalization of Razborov's result [10].

▶ **Definition 19.** We define a branching complexity measure μ_b to be a measure on functions which satisfies the following properties:
1. $\forall i, \mu_b(x_i) = \mu_b(\neg x_i) = 1$,
2. $\forall f, \mu_b(f) \geq 0$,
3. $\forall f, i, \mu_b(f \wedge x_i) + \mu_b(f \wedge \neg x_i) \leq \mu_b(f) + 2$,
4. $\forall f, g, \mu_b(f \vee g) \leq \mu_b(f) + \mu_b(g)$.

▶ **Definition 20.** Given a node v in a branching program, define $f_v(x)$ to be the function such that $f_v(x)$ is 1 if there is a path from some start node to v on input x and 0 otherwise. Note that for any start node s, $f_s = 1$.

▶ **Lemma 21.** *If μ_b is a branching complexity measure then for any branching program, the number of non-end nodes which it contains is at least*

$$\frac{1}{2}\left(\sum_{t:t \text{ is an end node}} \mu_b(f_t) - \sum_{s:s \text{ is a start node}} \mu_b(f_s)\right).$$

Proof. Consider what happens to $\sum_{t:t \text{ is an end node}} \mu_b(f_t) - \sum_{s:s \text{ is a start node}} \mu_b(f_s)$ as we construct the branching program. At the start, when we only have the start nodes and these are also our end nodes, this expression has value 0. Each time we merge end nodes together, this can only decrease this expression. Each time we branch off from an end node, making the current node a non-end node and creating two new end nodes, this expression increases by at most 2. Thus, the final value of this expression is at most twice the number of non-end nodes in the final branching program, as needed. ◀

▶ **Corollary 22.** *For any branching complexity measure μ_b and any function f, $\mu_b(f) \leq 130n$*

Proof. By Lemma 21 we have that for all $m \geq 1$, $\frac{m\mu_b(f) - m\mu_b(1)}{2} \leq m \cdot b_m(f)$. Using Theorem 6 and noting that $\mu_b(1) \leq 2$ we obtain that $\mu_b(f) \leq 130n$. ◀

Finally, we note that every submodular complexity measure μ_s is a branching complexity measure, so Corollary 22 is a slight generalization of Razborov's result [10] (though with a worse constant).

▶ **Definition 23.** A submodular complexity measure μ_s is a measure on functions which satisfies the following properties:
1. $\forall i, \mu(x_i) = \mu(\neg x_i) = 1$,
2. $\forall f, \mu(f) \geq 0$,
3. $\forall f, g, \mu_s(f \vee g) + \mu_s(f \wedge g) \leq \mu_s(f) + \mu_s(g)$.

▶ **Lemma 24.** *Every submodular complexity measure μ_s is a branching complexity measure.*

Proof. Note that

$$\mu_s(f \vee x_i) + \mu_s(f \wedge x_i) \leq \mu_s(f) + \mu_s(x_i)$$

and

$$\mu_s((f \vee x_i) \wedge \neg x_i) + \mu_s((f \vee x_i) \vee \neg x_i) = \mu_s(f \wedge \neg x_i) + \mu_s(1) \leq \mu_s(f \vee x_i) + \mu_s(\neg x_i).$$

Combining these two inequalities we obtain that

$$\mu_s(f \wedge \neg x_i) + \mu_s(1) + \mu_s(f \wedge x_i) \leq \mu_s(f) + \mu_s(x_i) + \mu_s(\neg x_i)$$

which implies that $\mu_s(f \wedge \neg x_i) + \mu_s(f \wedge x_i) \leq \mu_s(f) + 2 - \mu_s(1) \leq \mu_s(f) + 2$, as needed. ◀

7 Conclusion

In this paper, we showed that for any function f, there is a branching program computing a doubly exponential number of copies of f which has linear size per copy of f. This result shows that in the branching program model, any operation/function can be amortized with sufficiently many copies. This result also disproves a conjecture about nonuniform catalytic computation, rules out certain approaches for proving general lower space bounds, and gives a constructive analogue of Razborov's result [10] on submodular complexity measures.

However, the number of copies required in our construction is extremely large. A remaining open problem is to determine whether having a doubly exponential number of copies is necessary or there a construction with a smaller number of copies. Less ambitiously, if we believe but cannot prove that a doubly exponential number of copies is necessary, can we show that a construction with fewer copies would have surprising implications?

Acknowledgments. The author would like to thank Avi Wigderson and Venkatesh Medabalimi for helpful conversations.

References

1 Laszlo Babai, Noam Nisan, and Mario Szegedy. Multiparty Protocols, Pseudorandom Generators for Logspace, and Time-Space trade-offs. *Journal of Computer and System Sciences*, 45:204–232, 1992.

2 Harry Buhrman, Richard Cleve, Michal Koucký, Bruno Loff, and Florian Speelman. Computing with a full memory: catalytic space. In *46th Annual Symposium on the Theory of Computing (STOC 2014)*, pages 857–866, 2014. doi:10.1145/2591796.2591874.

3 Siu Man Chan and Aaron Potechin. Tight Bounds for Monotone Switching Networks via Fourier Analysis. *Theory of Computing*, 10:389–419, 2014. doi:10.1145/2213977.2214024.

4 Stephen Cook, Jeff Edmonds, Venkatesh Medabalimi, and Toniann Pitassi. Lower Bounds for Nondeterministic Semantic Read-Once Branching Programs. In *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*, volume 55 of *LIPIcs*, pages 36:1–36:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.ICALP.2016.36.

5 Vincent Girard, Michal Koucký, and Pierre McKenzie. Nonuniform catalytic space and the direct sum for space, 2015. URL: https://eccc.weizmann.ac.il/report/2015/138/.

6 Oleg Lupanov. The synthesis of contact circuits. *Doklady Akademii Nauk SSSR*, 119:23–26, 1958.

7 William Masek. A fast algorithm for the string editing problem and decision graph complexity. Master's thesis, MIT, 1976.

8 E. I. Nechiporuk. On a Boolean function. *Soviet Mathematics Doklady*, 7(4):999–1000, 1966.

9 Alexander Razborov. Lower bounds for deterministic and nondeterministic branching programs. In *Proceedings of the 8th International Symposium on Fundamentals of Computation Theory (FCT)*, volume 529 of *LNCS*, pages 47–60, 1991.

10 Alexander Razborov. On submodular complexity measures. In *Proceedings of the London Mathematical Society symposium on Boolean function complexity*, pages 76–83, 1992.

11 Claude Shannon. The synthesis of two-terminal switching circuits. *Bell System Technical Journal*, 28:59–98, 1949.

12 Dietmar Uhlig. On the synthesis of self-correcting schemes for functional elements with a small number of reliable elements. In *Mathematical Notes of the Academy of Sciences of the USSR*, volume 16, pages 558–562, 1974.

13 Dietmar Uhlig. Networks computing boolean functions for multiple input values. In *Proceedings of the London Mathematical Society Symposium on Boolean Function Complexity*, pages 165–173, 1992.

14 Ingo Wegener. On the complexity of branching programs and decision trees for clique functions. *Journal of the Association for Computing Machinery (JACM)*, 35(2):389–419, 1988. doi:10.1145/42282.46161.

15 Stanislav Zak. An exponential lower bound for one-time-only branching programs. In *Proceedings of the Mathematical Foundations of Computer Science (MFCF)*, pages 562–566, 1984.

Derandomizing Isolation in Space-Bounded Settings*

Dieter van Melkebeek[1] and Gautam Prakriya[2]

1 **University of Wisconsin-Madison, Madison, WI, USA**
`dieter@cs.wisc.edu`
2 **University of Wisconsin-Madison, Madison, WI, USA**
`prakriya@cs.wisc.edu`

Abstract

We study the possibility of deterministic and randomness-efficient isolation in space-bounded models of computation: Can one efficiently reduce instances of computational problems to equivalent instances that have at most one solution? We present results for the NL-complete problem of reachability on digraphs, and for the LogCFL-complete problem of certifying acceptance on shallow semi-unbounded circuits.

A common approach employs small weight assignments that make the solution of minimum weight unique. The Isolation Lemma and other known procedures use $\Omega(n)$ random bits to generate weights of individual bitlength $O(\log n)$. We develop a derandomized version for both settings that uses $O((\log n)^{3/2})$ random bits and produces weights of bitlength $O((\log n)^{3/2})$ in logarithmic space. The construction allows us to show that every language in NL can be accepted by a nondeterministic machine that runs in polynomial time and $O((\log n)^{3/2})$ space, and has at most one accepting computation path on every input. Similarly, every language in LogCFL can be accepted by a nondeterministic machine equipped with a stack that does not count towards the space bound, that runs in polynomial time and $O((\log n)^{3/2})$ space, and has at most one accepting computation path on every input.

We also show that the existence of somewhat more restricted isolations for reachability on digraphs implies that NL can be decided in logspace with polynomial advice. A similar result holds for certifying acceptance on shallow semi-unbounded circuits and LogCFL.

1998 ACM Subject Classification F.1.1 Models of Computation, F.1.2 Modes of Computation, F.1.3 Complexity Measures and Classes

Keywords and phrases Isolation lemma, derandomization, unambiguous nondeterminism, graph reachability, circuit certification

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.5

1 Introduction

Isolation is the process of singling out a solution to a problem that may have many solutions. It is used in algorithms with an algebraic flavor in order to prevent cancellations from happening. Examples include reductions of multivariate to univariate polynomial identity testing [39, 2] and recent approaches to the hamiltonicity problem [11, 29, 20, 19]. The process also plays an important role in the design of parallel algorithms, where it ensures that

* Research partially supported by the US National Science Foundation (grant CCF-1319822), the Foundation Sciences Mathématiques de Paris and the French Agence Nationale de la Recherche (grant ANR-13-BS02-0001-01 Compa), and the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant ERC-2014-CoG 648276 AUTAR).

32nd Computational Complexity Conference (CCC 2017).
Editor: Ryan O'Donnell; Article No. 5; pp. 5:1–5:32

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

the various parallel processes all work towards a single global solution rather than towards individual solutions that may not be compatible with one another. Both uses culminate in the asymptotically best known parallel algorithms for finding perfect matchings in graphs [47] and related problems [37, 1, 44]. A wide range of other algorithmic applications of isolation exist [58, 9, 10, 35, 56, 63, 50, 3, 46, 21, 59, 7, 27, 36, 31, 14, 38, 54, 12, 22, 32, 23, 45, 33]. In complexity theory isolation constitutes a key tool to show that in some computational models hard problems are no easier to solve on instances with unique solutions [16, and references further in this section].

Becoming more precise, let us define a computational (promise)[1] problem as a mapping $\Pi : X \mapsto 2^Y$ from an instance $x \in X$ to a set $\Pi(x)$ of solutions $y \in Y$, where x and y are strings that typically represent other types of objects. Given an instance $x \in X$, the decision version of Π asks to determine whether $\Pi(x)$ is nonempty. We denote by $L(\Pi)$ the set (language) of all instances $x \in X$ for which the decision is positive. The search version of Π asks to to produce a solution $y \in \Pi(x)$, or report that no solution exists. For example, for the NP-complete problem of Satisfiability, x represents a Boolean formula, and $\Pi(x)$ its satisfying assignments. For the NL-complete problem of Reachability, x represents a triple (G, s, t) consisting of a directed graph G, a start vertex s, and a target vertex t, and $\Pi(x)$ is the set of paths from s to t in G.

A nondeterministic machine M is said to *accept* Π (or $L(\Pi)$) if for every $x \in X$, M on input x has an accepting computation path if and only if $x \in \Pi$. We say that the machine M *decides* Π (or $L(\Pi)$) if M has an accepting computation path on every $x \in X$, and on each such path M outputs a bit indicating whether $\Pi(x) \neq \varnothing$. Note that the existence of a nondeterministic machine M that decides $L(\Pi)$ is equivalent to the existence of nondeterministic machines M_+ and M_- of the same complexity that accept $L(\Pi)$ and the complement of $L(\Pi)$, respectively. We say that M *computes* Π if it decides Π and on each accepting computation paths additionally outputs some $y \in \Pi(x)$ (which can depend on the path).

Within this framework we formalize the notion of isolation and distinguish between two types.

▶ **Definition 1** (Notions of isolation)**.** An isolation for a computational problem $\Pi : X \mapsto 2^Y$ is a mapping reduction f that transforms $x \in X$ into an "equivalent" instance $f(x) \in X$ with $|\Pi(f(x))| \leq 1$. A disambiguation is an isolation where equivalence requires that $\Pi(x)$ is empty if and only if $\Pi(f(x))$ is. A pruning is a disambiguation where equivalence additionally requires that $\Pi(f(x)) \subseteq \Pi(x)$.

Disambiguations are isolations geared towards decision problems. Prunings are isolations geared towards search problems. Actually, for search problems it suffices to have an intermediate notion, namely a recoverable disambiguation f, i.e, one for which there exists an efficient transformation f' that takes any solution $y \in \Pi(f(x))$ and turns it into a solution $f'(x, f(x), y) \in \Pi(x)$.

A closely related notion in the machine realm is that of unambiguity. A nondeterministic machine M is called *unambiguous* on an input x if it has at most one accepting computation path on input x. The machine is called unambiguous if it is unambiguous on every input x.

A common way to achieve isolation is by introducing a weight function $\omega : X \times Y \mapsto \mathbb{N}$ and restricting the set of solutions to those of minimum weight, in the hope that there is

[1] We use the prefix "promise" when we want to make it clear that the domain X of Π may be restricted, i.e., may not equal the set of all strings.

unique solution of minimum weight (or none in the case where there are no solutions). We use the following terminology.

▶ **Definition 2** (Min-isolation). Given $\omega : X \times Y \mapsto \mathbb{N}$, the min-weight of $x \in X$ is defined as

$$\omega(x) = \begin{cases} \min_{y \in \Pi(x)}(\omega(x,y)) & \text{if } \Pi(x) \neq \varnothing \\ \infty & \text{otherwise.} \end{cases}$$

We call ω min-isolating for x if there is at most one $y \in \Pi(x)$ with $\omega(x,y) = \omega(x)$.

In order to construct an actual isolation for Π, we need to express the restricted search on input x for a solution of weight $\mu = \omega(x)$ as an instance $f(x)$ of Π.

In many cases a suitable min-isolating weight function can be obtained by viewing the solutions y for a given instance x as subsets of a finite universe $U = U(x)$, assigning small weights $w(u) \in \mathbb{N}$ to the elements $u \in U$, and defining $\omega(x,y)$ as a linear combination of the weights $w(u)$ of the elements $u \in y$. In fact, the trivial linear combination (all coefficients 1) often suffices. If the linear combination is clear from context, we often abuse notation and use w in lieu of ω, e.g., writing $w(y)$ for $\omega(x,y)$, or $w(x)$ for $\omega(x)$, or applying the term "min-isolating" to w.

The known generic isolation procedures [60, 47, 18] are all *randomized*. A randomized isolation with success probability p is a randomized mapping reduction f that, on every instance $x \in X$, satisfies the defining requirements for an isolation on input x with probability at least p. In the min-isolation approach via a weight assignment to the underlying universe, randomness comes into play in the construction of the weight assignment. The following well-known mathematical fact (rephrased using our terminology) forms the basis.

▶ **Fact 3** (Isolation Lemma [47]). *Suppose that* $\Pi(x) \subseteq 2^U$ *and that* $\omega(x,y) = \sum_{u \in y} w(u)$ *for* $y \in \Pi(x)$. *For any positive integer* q, *if* $w : U \mapsto [q \cdot |U|]$ *is picked uniformly at random then* ω *is min-isolating for* x *with probability at least* $1 - 1/q$.

An important feature of the Isolation Lemma is that it keeps the range of the min-weight small, namely within $[c \cdot |U|^2]$. Once we have a min-isolating weight assignment of small range, we can further pick an integer μ uniformly at random within that range, and look for a solution $y \in \Pi(x)$ with $\omega(x,y) = \mu$. If μ happens to equal $\omega(x)$, there is a unique such y. The small range of the min-weight guarantees a reasonable probability of success p.

We can apply this process to Satisfiability with U denoting the set of variables of the formula x, and $q = 2$, say. The probability of success is $\Omega(1/n^2)$, where n denotes the number of variables of x. Since the weight restriction can be translated in polynomial time into a Boolean formula on the variables of the original formula, the resulting randomized isolation can be computed in polynomial time and is of the pruning type. The former implies that NP $\subseteq$ R$\cdot$PromiseUP [60].[2] Intuitively, the result means that, in the randomized time-bounded setting, having unique solutions does not make instances of NP-complete problems easier. Formally, R denotes the one-sided error (no false positives) probabilistic operator on classes $\mathcal{C}$ of languages: R$\cdot\mathcal{C}$ is the class of languages L for which there exists a constant $c \in \mathbb{N}$ and a language $C \in \mathcal{C}$ such that for all inputs x:

$$\begin{aligned} x \in L &\Rightarrow \Pr_\rho[\langle x, \rho \rangle \in C] \geq 1/n^c \\ x \notin L &\Rightarrow \Pr_\rho[\langle x, \rho \rangle \in C] = 0, \end{aligned}$$

[2] The original argument in [60] uses a different randomized isolation for Satisfiability; it has a success probability of $\Omega(1/n)$.

where ρ is picked uniformly at random from $\{0,1\}^{n^c}$, and n denotes the input length $|x|$. The operator extends to classes of promise problems in a natural way. PromiseUP represents the class of promise decision problems that can be accepted by nondeterministic polynomial-time machines that are unambiguous on every input satisfying the promise.

Isolation in Space-Bounded Settings. Gal and Wigderson [30] obtained a randomized isolation for Reachability by applying the Isolation Lemma in a similar fashion with the edges for the graph G as the universe U. Since the weighted reachability problem with polynomially bounded weights is also in NL, one can translate the weight restricted instance into an equivalent instance in logarithmic space, though on a graph with more vertices. This results in a randomized disambiguation with success probability $1/poly(n)$ that is computable in logarithmic space with two-way access to the random bits. (The disambiguation is recoverable in deterministic logspace, but is not a pruning.) It follows that $\mathrm{NL} \subseteq \mathrm{R} \cdot \mathrm{PromiseUL}$, where PromiseUL is the logspace equivalent of PromiseUP. Thus, in the randomized space-bounded setting, having unique solutions does not make instances of NL-complete problems easier.

Reinhardt and Allender [51] strengthened this result to $\mathrm{NL} \subseteq \mathrm{R} \cdot (\mathrm{UL} \cap \mathrm{coUL})$. The class UL consists of the problems in PromiseUL for which the promise holds for all inputs. In other words, UL is the class of languages accepted by unambiguous logspace machines. The significance of the strengthening is that within the class $\mathrm{R} \cdot (\mathrm{UL} \cap \mathrm{coUL})$ the probability of error can be reduced to exponentially small levels, allowing the randomness to be replaced by polynomial advice, i.e., $\mathrm{R} \cdot (\mathrm{UL} \cap \mathrm{coUL}) \subseteq (\mathrm{UL} \cap \mathrm{coUL})/\mathrm{poly}$. It follows that Reachability has a randomized disambiguation with exponentially small error that is computable in logspace with two-way access to the random bits, as well as a disambiguation that is computable in logspace with polynomial advice.

The construction in [51] needs a stronger property of the weight assignment w than merely being min-isolating on the given input (G, s, t). It requires w to be min-isolating for G, i.e., min-isolating for (G, s, t) for *all* choices of vertices s and t. By setting $q = 2n^2$ in the Isolation Lemma, a union bound guarantees that with probability at least 50%, a random weight assignment $w : E \mapsto [2n^2 m]$ is min-isolating for any given graph $G = (V, E)$ with n vertices and m edges. The randomness in $\mathrm{NL} \subseteq \mathrm{R} \cdot (\mathrm{UL} \cap \mathrm{coUL})$ is still only used to generate random weight assignments. The new ingredients in [51] that enable the strengthening from $\mathrm{R} \cdot \mathrm{PromiseUL}$ to $\mathrm{R} \cdot (\mathrm{UL} \cap \mathrm{coUL})$ are unambiguous logspace machines to (i) decide whether or not a given weight assignment is min-isolating for a given graph G, and (ii) compute the min-weight $w(G, s, t)$ under a given min-isolating weight assignment w.

Gal and Wigderson [30] and Reinhardt and Allender [51] developed analogous results for the complexity class LogCFL, where the role of Reachability is taken over by the problem Circuit Certification of finding a certificate that a given Boolean circuit accepts a given input. We refer to Section 3.1 for the definition of a certificate and for background, and defer further discussion of this setting to that section.

Derandomizing Isolation. The number of random bits needed for an application of the Isolation Lemma as stated is $\Theta(n \log(qn))$, namely $\Theta(\log(qn))$ bits for each of the $n \doteq |U|$ elements of the universe U. In order to develop variants that require fewer random bits, we introduce the notion of a *weight assignment generator*, which can be viewed as a structured form of a pseudorandom generator geared towards the setting of the Isolation Lemma. Whereas a pseudorandom generator is parameterized by the desired length of the

pseudorandom sequence, a weight assignment generator is parameterized by the desired domain D of the weight assignments.

▶ **Definition 4** (Weight assignment generator)**.** A weight assignment generator Γ for a family of domains $\mathcal{D}$ is a family of mappings $(\Gamma_D)_{D\in\mathcal{D}}$ such that Γ_D takes a string $\sigma \in \{0,1\}^{s(D)}$ for some function $s : \mathcal{D} \mapsto \mathbb{N}$, and maps it to a weight assignment $w : D \mapsto \mathbb{N}$. We say that w is chosen uniformly at random from Γ_D if it is obtained as $w = \Gamma_D(\sigma)$ where σ is chosen uniformly at random from $\{0,1\}^{s(D)}$.

The family of domains $\mathcal{D}$ in Definition 4 is usually indexed by one or more integer parameters, in which case we also index Γ that way. For example, for a derandomization of the Isolation Lemma we can equate the universe U with $[|U|] \doteq \{1, 2, \ldots, |U|\}$ by ordering the elements of U in some way, e.g., lexicographically. We can then choose $\mathcal{D} = (D_n)_{n\in\mathbb{N}}$ with $D_n \doteq [n]$, and write Γ_n for Γ_{D_n}.

The relevant characteristics of a weight assignment generator are the following:

- The seed length $s(D)$, which is the number of random bits we need when we pick a weight assignment from Γ_D uniformly at random.
- The maximum weight assigned by Γ_D, the logarithm of the maximum weight is called the *bitlength of the generator*. A bound on the weights is sometimes also used as a parameter indexing the generator (in addition to the domain D).
- The computational complexity of Γ, by which we mean the complexity of the deciding, on input the parameters p, $\sigma \in \{0,1\}^s$, $z \in D$, $i \in \mathbb{N}$, and $b \in \{0,1\}$, whether the ith bit of $w(z)$ for $w = \Gamma_p(\sigma)$ is b.

The Isolation Lemma can be viewed as a generic weight assignment generator (for the family of domains $([n])_{n\in\mathbb{N}}$) that has seed length $O(n \log(qn))$, bitlength $O(\log n)$, and trivial complexity. By allowing weights that are polynomially larger than in the Isolation Lemma, one can achieve seed length $O(\log(qn) + \log(|\Pi(x)|))$, which is provably optimal for a generic $\Pi(x)$ [18]. In our setting this yields seed length $O(n)$ and bitlength $O(\log n)$. In order to do better, one needs to exploit the specifics of the set systems. Doing so generically in the time-bounded setting seems difficult. There are implications from derandomizing the Isolation Lemma for generic $\Pi(x)$ of small circuit complexity to circuit lower bounds of various sorts [6], and vice versa [40]. The circuit lower bounds are arguably reasonable but have been open for a long time. There may be ways to obtain deterministic or derandomized isolations other than by derandomizing the Isolation Lemma, but for Satisfiability the existence of a deterministic polynomial-time pruning implies that NP $\subseteq$ P/poly. In fact, the collapse follows from the existence of a randomized polynomial-time pruning that has success probability $p > 2/3$ [26].

In the space-bounded setting there is more hope to obtain unconditional derandomizations. An implication from lower bounds to derandomization still holds: If there exists a problem in DSPACE(n) that requires Boolean circuits of linear-exponential size, then there exists a logspace computable weight assignment generator with seed length and bitlength $O(\log n)$ [40, 4]. There is no known result showing that deterministic isolations in the space-bounded setting imply circuit (or branching program) lower bounds that are open. Moreover, unconditional results already exist for certain restricted classes of digraphs. For Reachability on directed planar grid graphs, min-isolating weight assignments of bitlength $O(\log n)$ are known to be computable in deterministic logspace [13]. Those assignments have been used to construct disambiguations that are logspace computable and logspace recoverable for larger classes of graphs [13, 57, 43].

There have also been related successes for isolating PerfectMatching, the problem of deciding/finding perfect matchings in graphs, restricted to certain special types of graphs [24, 25, 5]. Recently, Fenner, Gurjar, and Thierauf [28] constructed a weight assignment generator with seed length and bitlength $O((\log n)^2)$ that is computable in logspace and that produces a min-isolating weight assignment for PerfectMatching on a given bipartite graph with probability at least $1-\log(n)/n$. This allowed them to prove that PerfectMatching on bipartite graphs has logspace-uniform circuits of polylogarithmic depth and quasi-polynomial size.

Main Results. We present positive and negative results regarding the possibility of derandomized isolations for Reachability and for Circuit Certification.

The crux for our positive results are logspace min-isolating weight assignment generators with seed length and bitlength $O((\log n)^{3/2})$. We actually shift the paradigm a bit – we assign weights to the (internal) *vertices* rather than to the edges. This is not an essential difference,[3] but it facilitates a natural iterative/recursive approach towards the construction of the weight assignment, and allows for a cleaner and unified treatment.

Recall that in the context of Reachability we call a weight assignment w min-isolating for an instance (G, s, t) if G has at most one path from s to t of minimum weight under w. For technical reasons we only consider the restriction of Reachability to *layered* digraphs G.

▶ **Theorem 5.** *There exists a weight assignment generator $\Gamma^{(\mathrm{reach})} = (\Gamma^{(\mathrm{reach})}_{n,d})_{n,d\in\mathbb{N}}$ that is computable in space $O(\log n)$ and has seed length and bitlength $O(\sqrt{\log d}\log n)$ such that for every layered digraph G of depth d with n vertices*

$$\Pr_w[w \text{ is min-isolating for } G] \geq 1 - 1/n,$$

where w is chosen uniformly at random from $\Gamma^{(\mathrm{reach})}_{n,d}$.

The domain underlying $\Gamma^{(\mathrm{reach})}_{n,d}$ is $[n] \times [\![d]\!] \doteq \{1, 2, \ldots, n\} \times \{0, 1, 2, \ldots, d\}$. We refer to Section 2.1 for more details.

We use Theorem 5 to derive the following isolation result for NL, where the notation $\mathrm{UTISP}(t, s)$ stands for the class of languages accepted by unambiguous nondeterministic machines that run in time t and space s.

▶ **Theorem 6.** $\mathrm{NL} \subseteq \mathrm{UTISP}(\mathrm{poly}(n), (\log n)^{3/2})$.

In words: Every language in NL can be accepted by a nondeterministic machine that runs in polynomial time and $O((\log n)^{3/2})$ space, and has at most one accepting computation path on every input.

Theorem 6 should be contrasted with the most space efficient simulation of NL on *deterministic* machines, which is given by Savitch's Theorem [53]: $\mathrm{NL} \subseteq \mathrm{DSPACE}((\log n)^2)$. That simulation does not run in polynomial time. In fact, the best upper bound on the running time is the one for generic computations in $\mathrm{DSPACE}((\log n)^2)$, namely $n^{O(\log n)}$. Reachability can be solved in linear time and space using depth-first search or breadth-first search. The smallest known space bound for an algorithm that decides Reachability in polynomial time is only slightly sublinear, namely $n/2^{\Theta(\sqrt{\log n})}$ [8].

On the "negative" side we show:

[3] We could alternately assign the weight of a vertex to each of its outgoing edges without affecting the total weight of any solution.

▶ **Theorem 7.** *Either one of the following hypotheses implies that* $\mathrm{NL} \subseteq \mathrm{L/poly}$*:*

1. Reachability *on layered digraphs has a logspace pruning.*
2. Reachability *on layered digraphs has a logspace weight function* ω *that is min-isolating, and there exists a logspace function* μ *such that* $\mu(x)$ *equals the min-weight* $\omega(x)$ *of* x *under* ω *on positive instances* x*.*[4]

In fact, the conclusion holds even if the algorithms are randomized, as long as the probability of success exceeds $\frac{2}{3} + \frac{1}{\mathrm{poly}(n)}$ *and the algorithms run in logspace when given two-way access to the random bits.*

It is not clear to us that Theorem 7 should be viewed as a roadblock towards reducing the seed length and bitlength in Theorem 5 from $O((\log n)^{3/2})$ down to $O(\log n)$, and thereby show that NL = UL. Regarding the first part of Theorem 7, none of the known randomized isolations for Reachability are of the pruning type. This is because they map an instance $x \doteq (G, s, t)$ to an instance $f(x)$ where the underlying graph contains more vertices than G, which makes it impossible to meet the pruning requirement that $\Pi(f(x)) \subseteq \Pi(x)$.

The corresponding results for Circuit Certification and the complexity class LogCFL are stated in Section 3 (positive) and Section 4 (negative).

Techniques. The crux for our positive results is an iterative/recursive construction of a min-isolating weight assignment generator Γ. In both settings there are $\Theta(\log n)$ levels of recursion. In the case of Reachability the subproblems at the kth level correspond to the subgraphs induced by blocks of 2^k successive layers of G. In the case of Circuit Certification the kth level corresponds to the kth level of AND gates of the given circuit.

We develop several methods to build out of a min-isolating weight assignment w_k at the kth level, a min-isolating weight assignment w_{k+1} at the $(k+1)$st level. The methods represent different trade-offs between the seed length and the bitlength. Our starting point is two simple constructions, namely one based on shifting, and one based on universal families of hash functions. The shifting approach does not need any randomness at all but yields bitlength $\Theta((\log n)^2)$. Hashing yields the smaller bitlength $O(\log n)$ but needs $\Theta((\log n)^2)$ random bits. Either one of those simple approaches on its own is sufficient to establish weaker versions of our positive results, namely where the randomness or space bound is increased from $O((\log n)^{3/2})$ to $O((\log n)^2)$, i.e.,

$$\mathrm{NL} \subseteq \mathrm{UTISP}(\mathrm{poly}(n), (\log n)^2). \tag{1}$$

The $\Theta((\log n)^2)$ bits of randomness in the hashing-based approach are composed of $\Theta(\log n)$ bits to describe a fresh hash function at each of the $\Theta(\log n)$ levels of recursion. The reason one needs a fresh hash function at each level is to avoid potential stochastic dependencies. We show how to use shifting to preclude the existence of such dependencies, allowing us to reuse the same hash function at $\Theta(\sqrt{\log n})$ levels. This combination of shifting and hashing balances the seed length and bitlength to $\Theta((\log n)^{3/2})$ each, and yields Theorem 5 and its counterpart for Circuit Certification.

For Theorem 6 and its counterpart for LogCFL we need to get rid of the randomness completely. We could do so by exhaustively trying all random seeds, and employing the unambiguous logspace machine of [51] to select one that yields a min-isolating weight assignment. However, given that the number of random bits is $\Theta((\log n)^{3/2})$, an exhaustive

[4] If $\mu(x) = \omega(x)$ on all instances x, we can easily decide Reachability in logspace as there exists a path from s to t in G if and only if $\omega(G, s, t) < \infty$.

search would require time $n^{\Theta(\sqrt{\log n})}$. In order to do better, we exploit the structure of the randomness – it consists of $\Theta(\sqrt{\log n})$ hash functions requiring $\Theta(\log n)$ random bits each. Using the unambiguous logspace machines from [51] this allows us to pick the hash functions one by one, maintaining the invariant that the resulting weight assignments are min-isolating for the corresponding levels, and then use the final assignment to decide reachability unambiguously. As we can cycle through all possibilities for a hash function at a given level in polynomial time, this yields a full derandomization running in polynomial time and space $O((\log n)^{3/2})$.

The "negative" results, Theorem 7 and its counterpart for Circuit Certification, follow along the lines of the argument for a similar result from [26] in the time-bounded setting. The first part is the space-bounded equivalent of the main result in [26]; it suffices to verify that the argument from the time-bounded setting carries over to the space-bounded setting. The second part does not have a counterpart in [26] but follows from a similar argument and some additional observations.

Related Papers. There is a remarkable correspondence in terms of statements and high-level approach between Theorem 6 and the result by Saks and Zhou [52] that BPL $\subseteq$ DSPACE$((\log n)^{3/2})$. Both have a recursive structure, use hashing,[5] need to get rid of stochastic dependencies so as to enable the reuse of the same hash function at multiple levels of recursion, exploit the leeway created by the discrepancy between the randomness and processing space (bitlength), and ultimately balance them to $\Theta((\log n)^{3/2})$ bits each. In contrast to [52], we do obtain the equivalent of a pseudorandom generator. As another contrast we are able to improve the running time to polynomial, which remains open in the case of BPL [15]. Our high-level approach for the improvement is similar to the one for the improvement from BPL $\subseteq$ DSPACE$((\log n)^2)$ in [48] to BPL $\subseteq$ DTISP$(\mathrm{poly}(n), (\log n)^2)$ in [49].

The recent derandomization results for PerfectMatching on bipartite graphs [28] and for polynomial identity testing (PIT) for read-once arithmetic branching programs [2] also employ a combination of hashing and shifting but no balancing. They need $O((\log n)^2)$ random bits as opposed to our $O((\log n)^{3/2})$. It is an open question whether our approach can be used to reduce the number of random bits in those settings. This question is related to the reduction from multivariate (multilinear) PIT to univariate PIT based on isolation: If $w : [n] \mapsto \mathbb{N}$ is a weight assignment to the variables that is min-isolating for the monomials that occur in a nonzero n-variate polynomial $P(x_1, x_2, \ldots, x_n)$, then the substituted polynomial $Q(t) \doteq P(t^{w(1)}, t^{w(2)}, \ldots, t^{w(n)})$ remains nonzero.

Recently, Kalampalli and Tewari [34] independently proved the weaker inclusion (1) that follows from either of our starting points (the pure shifting approach that needs no randomness and bitlength $\Theta((\log n)^2)$, and the pure hashing approach that needs $\Theta((\log n)^2)$ random bits and yields bitlength $O(\log n)$). In their construction both quantities are $\Theta((\log n)^2)$.

Very recently, Krishnan and Limaye [42] posted a report on ECCC in which they independently prove part 1 of our "negative" results (Theorem 7 and its counterpart for LogCFL), which follow from a space-bounded rendering of the main argument in [26].[6]

[5] [52] does so via Nisan's pseudorandom generator [48].

[6] The current version of the report claims that the arguments also rely on [51], but the authors agree that [51] is not needed there (personal communication).

Organization. In Section 2 we derive our positive results for Reachability and NL. The results essentially also follow as corollaries to the corresponding results for Circuit Certification and LogCFL, which we prove from scratch in Section 3. This organization allows us to develop our ideas in the more familiar setting of Reachability and NL in a gradual and somewhat informal way, and suffice with a formal proof without much intuition in the more general setting of Circuit Certification and LogCFL. In Section 4 we present our negative results for both settings.

2 Reachability and NL

In this section we develop our min-isolating weight assignment generator for Reachability (Theorem 5), and derive our positive isolation result for NL (Theorem 6).

2.1 Weight Assignment Generator

Recall the notion of min-isolation in the context of Reachability:

▶ **Definition 8** (Min-isolating weight assignment for Reachability)**.** Let $G = (V, E)$ be a digraph. A weight assignment for G is a mapping $w : V \mapsto \mathbb{N}$. The weight $w(P)$ of a path P in G is the sum of $w(v)$ over all vertices v on the path. For $s, t \in V$, $w(G, s, t)$ denotes the minimum of $w(P)$ over all paths from s to t, or ∞ if no such path exists. The weight assignment w is min-isolating for (G, s, t) if there is at most one path P from s to t with $w(P) = w(G, s, t)$. For $A \subseteq V \times V$, w is min-isolating for (G, A) if w is min-isolating for (G, s, t) for each $(s, t) \in A$. We call w min-isolating for G if w is min-isolating for $(G, V \times V)$.

We restrict attention to *layered* digraphs. A layered digraph $G = (V, E)$ of depth d consists of $d + 1$ layers of vertices such that edges only go from one layer to the next. More formally, with $n \doteq |V|$ we have that $V \subseteq [n] \times [\![d]\!] \doteq \{1, 2, \ldots, n\} \times \{0, 1, 2, \ldots, d\}$ and $E \subseteq \cup_{i \in [d]} (V_{i-1} \times V_i)$. We denote by $V_i \doteq V \cap [n] \times \{i\}$ the ith layer of G.

In fact, we only need to consider layered digraphs of depths that are powers of two. For $d = 2^\ell$ with $\ell \in \mathbb{N}$, and $k \in [\![\ell]\!]$, such a digraph can be viewed as consisting of $d/2^k = 2^{\ell-k}$ consecutive blocks of depth 2^k, where the ith block is the subgraph induced by the vertices in layers $(i-1)2^k$ through $i2^k$, i.e., $\cup_{j=(i-1)\cdot 2^k}^{i \cdot 2^k} V_j$.

We need to design a randomness efficient process that, given $d = 2^\ell$ and n, generates small weight assignments $w : [n] \times [\![d]\!] \mapsto \mathbb{N}$ that are min-isolating for any layered digraph $G = (V, E)$ of depth d on n vertices with high probability. Note that the use of the domain $[n] \times [\![d]\!]$ rather than merely $[n]$ enables the weight assignment to depend on the layer a vertex is in.

Iterative Approach. Given the recursive nesting structure of the blocks, there is a natural iterative/recursive approach towards the construction of w, based on the following simple observation:

> A min-weight path from s to t that passes through a vertex u is the concatenation of a min-weight path from s to u and a min-weight path from u to t.

We present an iterative (i.e., bottom-up) version, where in the kth iteration we try to construct a weight assignment w_k that is min-isolating for each block of depth 2^k and only assigns nonzero weights to the vertices that are internal to those blocks, i.e., to $V \setminus \cup_{i=0}^{2^{\ell-k}} V_{i \cdot 2^k}$.

We start with $w_0 \equiv 0$, and end with $w = w_\ell$. Here is how we move from w_k to w_{k+1} in iteration $k + 1$ for $k \in [\![\ell - 1]\!]$. Consider a block B of depth 2^{k+1}. It consists of two consecutive

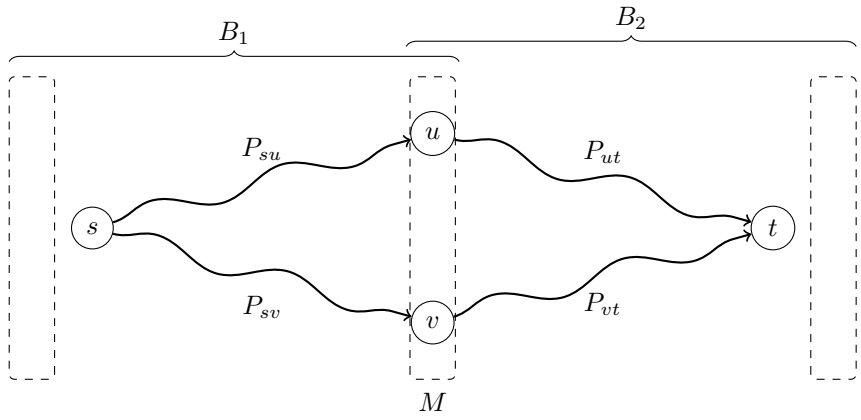


Figure 1

blocks B_1 and B_2 of depth 2^k that have the middle layer M of B in common (see Figure 1). The assignment w_k gives weights to all vertices of B except the initial layer, the middle layer M, and the final layer. We construct the assignment w_{k+1} by extending w_k, i.e., w_{k+1} keeps the values of w_k on the layers internal to B_1 or B_2, and additionally assigns weights to the vertices in M. We refer to the union of the middle layers M over all blocks of depth 2^{k+1} as the set L_{k+1} of vertices at level $k+1$, i.e.,

$$L_{k+1} \doteq \cup_{\text{odd } i \in [2^{\ell-k}]} V_{i \cdot 2^k}. \tag{2}$$

The new weights are assigned so as the maintain the invariant – Assuming that w_k is min-isolating for B_1 and B_2 individually, we want to make sure that w_{k+1} is min-isolating for all of B. Consider two vertices s and t in B such that s appears in an earlier layer than t.

- If t is internal to B_1 then w_{k+1} is min-isolating for (B, s, t) no matter how w_{k+1} assigns weights to M. This follows from the hypothesis and the fact that w_{k+1} and w_k agree on the vertices of B_1 other than M. The case where s is internal to B_2 is similar.
- Otherwise, s belongs to B_1 and t belongs to B_2. In that case every path from s to t has to cross layer M. We claim that among the paths (if any) that cross M in a fixed vertex v, there is a unique one of minimum weight with respect to w_{k+1}, say P_v. This follows from the above observation, the hypothesis, and the fact that w_{k+1} and w_k agree on the vertices of B other than M. Indeed, any such path P_v is the concatenation of a path P_{sv} in B_1 from s to v, and a path P_{vt} in B_2 from v to t. Since $w_{k+1}(P_v) = w_k(P_v) + w_{k+1}(v) = w_k(P_{sv}) + w_k(P_{vt}) + w_{k+1}(v)$, both P_{sv} and P_{vt} need to be min-weight with respect to w_k. By hypothesis, both those min-weight paths are uniquely determined, whence so is P_v.
  Thus, in order to guarantee that w_{k+1} is min-isolating for (B, s, t), it suffices to ensure that for all vertices $u, v \in M$ that are on a path from s to t,

$$\mu_k(s,u) + \mu_k(u,t) + w_{k+1}(u) \neq \mu_k(s,v) + \mu_k(v,t) + w_{k+1}(v), \tag{3}$$

  where $\mu_k(s,t) \doteq w_k(G,s,t)$ denotes the minimum weight of a path from s to t under w_k, or ∞ if no such path exists. We refer to condition (3) as a *disambiguation requirement*. See Figure 1 for an illustration.

We now consider three ways to meet the disambiguation requirements: shifting, hashing, and a combination of both. For each construction we track:

- the number R_k of random bits that w_k needs, and
- the maximum weight W_k of paths in G under w_k.

The quantity $R \doteq R_\ell$ corresponds to the seed length of the weight assignment generator Γ. The logarithm of the quantity $W \doteq W_\ell$ equals the bitlength of Γ up to an additive term of $O(\log d)$. As we will see in Section 2.2, the simulations of NL on unambiguous machines that we obtain via Γ run in space $O(R + \log(W) + \log(n))$. Thus, our aim is to minimize the quantity $R + \log(W)$ up to constant factors. We will ultimately succeed in making it as small as $O((\log n)^{3/2})$. Ideally, we would like to reduce it further to $O(\log n)$ so as to establish NL $\subseteq$ UL.

Shifting. For $v \in M \subseteq L_{k+1}$ we set $w_{k+1}(v) = \text{index}(u) \cdot b$, where b is an integer that exceeds W_k, and index is an injective function from M to $\mathbb{N}$. As the vertices in V are represented as pairs $(i,j) \in [\![d]\!] \times [n]$ and all vertices in M have the same first component, we can simply use the projection $(i,j) \mapsto j$ as the index function. This guarantees distinct values for the two sides of (3) for different u and v, irrespective of the values of $\mu_k(s,u) + \mu_k(u,t)$ and $\mu_k(s,v) + \mu_k(v,t)$. In terms of binary representations, if b is a power of 2, this construction can be interpreted as shifting the index function into a region of the binary representation that has not been used before.

We have that $R_{k+1} = R_k$ and $W_{k+1} \leq W_k + 2^{\ell-k-1} \cdot n \cdot b \leq (dn+1)(W_k+1) - 1$. When we use shifting at all levels, we end up with $R = 0$ and $W \leq (dn+1)^\ell = n^{O(\log n)}$, so $R + \log(W) = O((\log n)^2)$.

Hashing. When $w_{k+1}(u)$ and $w_{k+1}(v)$ are picked uniformly at random from a sufficiently large range, independently from each other and from the values $\mu_k(s,u) + \mu_k(u,t)$ and $\mu_k(s,v) + \mu_k(v,t)$, the disambiguation requirement (3) holds with high probability. We make use of *universal hashing* to obtain the random values we need using few random bits, and in particular of the following well-known family and property. We cast the notion in terms of a weight assignment generator with a bound on the weights as an additional parameter.

▶ **Fact 9** (Universal hashing [17]). *There exists a logspace computable weight assignment generator $(\Gamma^{(\text{hashing})}_{m,r})_{m,r \in \mathbb{N}}$ with seed length $s(m,r) = O(\log(mr))$ such that $\Gamma^{(\text{hashing})}_{m,r}$ produces functions $h : [m] \mapsto [r]$ with the following property: For every $u, v \in [m]$ with $u \neq v$, and every $a, b \in \mathbb{N}$*

$$\Pr_h[a + h(u) = b + h(v)] \leq 1/r, \tag{4}$$

where h is chosen uniformly at random from $\Gamma^{(\text{hashing})}_{m,r}$.

We identify $D \doteq [\![d]\!] \times [n]$ with $[m] = [d \cdot n]$ in a natural way. If we pick $h : D \mapsto [r]$ uniformly at random from $\Gamma^{(\text{hashing})}_{m,r}$ and set $w_{k+1} = h$ on L_{k+1}, (4) guarantees that each individual disambiguation requirement (3) holds with probability at least $1 - 1/r$. As there are at most n^4 choices for (s,t,u,v), a union bound shows that all disambiguation conditions are met simultaneously with probability at least $1 - n^4/r$. It suffices to pick r as a sufficiently large polynomial in n in order to guarantee high success probability. In particular, $r = n^6$ suffices for probability of success at least $1 - 1/n^2$.

Based on the charateristics of the family of hash functions $\Gamma^{(\text{hashing})}$ from Fact 9, we have that $R_{k+1} = R_k + O(\log(dnr)) = R_k + O(\log n)$ and $W_{k=1} \leq W_k + 2^{\ell-k-1} \cdot r \leq W_k + dr = W_k + n^{O(1)}$. When we use a fresh uniform sample $h = h_k$ from $\Gamma^{(\text{hashing})}$ for each iteration $k \in [\ell]$, we end up with $R = O(\ell \log(n)) = O((\log n)^2)$, and $W = \ell \cdot n^{O(1)} = n^{O(1)}$, so $R + \log(W) = O((\log n)^2)$ again.

Combined Approach. The shifting approach is ideal in terms of the amount of randomness R but leads to weights that are too large. The hashing approach is ideal in terms of the bound W on the path weights but requires too many random bits. We now combine the two approaches so as to balance R and $\log(W)$. The construction can be viewed as incorporating shifting into the hashing approach, or vice versa. Our presentation follows the former perspective.

In order to reduce the number of random bits in the hashing approach, we attempt to employ the same hash function h in multiple successive iterations, say iterations $k+1$ through k', going from w_k to $w_{k'}$. This does not work as such because the minimum path weights in the disambiguation requirements (3) for iterations above $k+1$ depend on h, and we cannot guarantee the bound (4) if a or b depend on h. However, the influence of the choice of h on those minimum path weights is limited. More specifically, in iteration $k+2$ we have that for any s and t that belong to the same block of depth 2^{k+1}

$$\mu_k(s,t) \le \mu_{k+1}(s,t) \le \mu_k(s,t) + r. \tag{5}$$

The first inequality follows because $w_{k+1} \ge w_k$. The second one follows by considering a minimum-weight path P from s to t under w_k and realizing that

$$\mu_{k+1}(s,t) \le w_{k+1}(P) = w_k(P) + h(v) = \mu_k(s,t) + h(v) \le \mu_k(s,t) + r,$$

where v is the unique vertex in $P \cap L_{k+1}$.

Let b be a power of two to be determined later. Equation (5) implies that $\mu_k(s,t)$ and $\mu_{k+1}(s,t)$ are the same after truncating the $\log b$ lowest-order bits, i.e., $\lfloor \mu_k(s,t)/b \rfloor = \lfloor \mu_{k+1}(s,t)/b \rfloor$, unless adding r to $\mu_k(s,t)$ results in a carry into bit position $\log b$ (the position corresponding to the power $2^{\log b} = b$). Suppose we can prevent such carries from happening. Conceptually, in iteration $k+2$ we can then apply the hashing approach with the *same* hash function h as in iteration $k+1$ provided we use the *truncated* values $\mu'_{k+1}(s,t) \doteq \lfloor \mu_k(s,t)/b \rfloor = \lfloor \mu_{k+1}(s,t)/b \rfloor$ as the minimum path weights. Indeed, since the values μ'_{k+1} are independent of h, (4) in Fact 9 shows that the disambiguations requirements with respect to μ'_{k+1}, i.e.,

$$\mu'_{k+1}(s,u) + \mu'_{k+1}(u,t) + h(u) \ne \mu'_{k+1}(s,v) + \mu'_{k+1}(v,t) + h(v), \tag{6}$$

hold with high probability. Undoing the truncation, (6) implies that

$$\mu_{k+1}(s,u) + \mu_{k+1}(u,t) + h(u) \cdot b \ne \mu_{k+1}(s,v) + \mu_{k+1}(v,t) + h(v) \cdot b.$$

Thus, by setting $w_{k+2}(v) = h(v) \cdot b$ for $v \in L_{k+2}$ we realize the actual disambiguation requirements for iteration $k+2$ with high probability *in conjunction* with the disambiguation requirements for iteration $k+1$. The setting of w_{k+2} on L_{k+2} can be interpreted as using a shifted version of the same hash function h instead of h itself.

We can repeat the process for iterations $k+3$ through k'. In iteration $k+i$, the bound (5) becomes

$$\mu_k(s,t) \le \mu_{k+i-1}(s,t) \le \mu_k(s,t) + r \cdot (b^{i-2} + 2b^{i-3} + \ldots + 2^{i-2}) \le \mu_k(s,t) + 2rb^{i-2},$$

where the last inequality assumes that $b \ge 4$. We set $w_{k+i}(v) = h(v) \cdot b^{i-1}$ for $v \in L_{k+i}$, and achieve our goal if h satisfies the disambiguation requirements

$$\lfloor \mu_k(s,u)/b^{i-1} \rfloor + \lfloor \mu_k(u,t)/b^{i-1} \rfloor + h(u) \ne \lfloor \mu_k(s,v)/b^{i-1} \rfloor + \lfloor \mu_k(v,t)/b^{i-1} \rfloor + h(v) \tag{7}$$

for all appropriate choices of s, t, u, v. Equation (4) in Fact 9 and a union bound show that the requirements (7) are all met simultaneously by the same hash function h for all iterations $k+1$ through k' with probability at least $1 - \Delta/n^2$ for $r = n^6$, where $\Delta \doteq k' - k$.

In iteration $k+2$ we made the assumption that there are no carries into position $\log b$ when adding r to the values μ_k. More generally, in iteration $k+i$, we assumed there are no carries into position $(i-1)\cdot\log b$ when adding $2rb^{i-2}$ to the values μ_k. The assumption holds if $b \geq 4r$ and the values μ_k have a 0 in the position right before each of the positions $(i-1)\cdot\log b$. We can maintain the latter condition as an invariant throughout the construction by setting $b = O(r)$ sufficiently large.

Alternately, $b \geq 2r$ is enough to ensure that the carries are no larger than 1. We can handle such carries by strengthening the disambiguation requirements (7) and impose that the left-hand side and right-hand side are not just distinct but are separated by a small constant. This only involves a constant factor more of applications of (4) in the union bound, and guarantees that the values remain distinct after undoing the truncation. In fact, it suffices to require for all $i \in [k' - k]$ that

$$\lfloor (\mu_k(s,u) + \mu_k(u,t))/b^{i-1} \rfloor + h(u) \notin \lfloor (\mu_k(s,v) + \mu_k(v,t))/b^{i-1} \rfloor + h(v) + \{-1, 0, 1\}$$

for some $b \geq 4r$. We refer to the formal proof of Lemma 15 in Section 3.2 for more details (in the setting of Circuit Certification instead of Reachability), in particular to the argument for Claim 16.

We obtain the following characteristics: $R_{k'} = R_k + O(\log(dnr)) = R_k + O(\log n)$ and $W_{k'} \leq W_k + 2^{\ell - k'} \cdot 2rb^{\Delta-1} \leq W_k + drb^{\Delta-1} = W_k + O(n^\Delta)$, where $\Delta \doteq k' - k$.

Final Construction. Starting from $w_0 \equiv 0$, for any $\Delta \in [\ell]$ we can apply the combined construction ℓ/Δ times consecutively to obtain $w = w_\ell$. Each application uses a fresh hash function to bridge the next Δ levels. The setting $\Delta = 1$ corresponds to the pure hashing approach, and the setting $\Delta = \ell$ essentially to the pure shifting approach.[7] We can interpolate between the parameters of the pure shifting and pure hashing approaches by considering intermediate values of Δ. We have $R = O(\Delta/\ell \cdot \log n)$ and $W = O(\Delta/\ell \cdot n^\Delta)$, so $R + \log(W) = O((\Delta/\ell + \Delta)\log n)$. The latter expression is minimized up to constant factors when $\Delta/\ell = \ell$, i.e., when $\Delta = \sqrt{\ell}$, yielding a value of $R + \log(W) = O(\sqrt{\ell}\log n) = O(\sqrt{\log d}\log n) = O((\log n)^{3/2})$.

The above construction yields a weight assignment generator $\Gamma^{(\text{reach})}$ that is indexed by the number of vertices n and the depth d, with $D_{n,d} \doteq [n] \times [\![d]\!]$ as the domain for the weight functions given by $\Gamma^{(\text{reach})}_{n,d}$. The construction works for any d that is a power of 2 and any $n \in \mathbb{N}$, and has the properties stated in Theorem 5. We already analyzed the seed length and bitlength. For any given layered digraph of depth d on n vertices, the failure probability at each level of the construction is at most $1/n^2$. As there are $\ell \doteq \log d \leq \log n$ levels, the overall failure probability is at most $\log(n)/n^2 \leq 1/n$. The logspace computability follows from the logspace computability of the underlying universal family of hash functions and the fact that iterated addition is in logspace (see, e.g., [62]).

Values of d that are not powers of 2 can be handled by first extending the given layered digraph G with identity matchings (for each i connect the ith gate in the next layer with the ith gate in the previous layer) until the depth reaches a power of 2, and then applying the above construction.

[7] In this setting the hash function h from the "combined" approach can be replaced by an index function.

This concludes a somewhat informal proof of Theorem 5. Section 3.2 contains a more formal proof (in the setting of Circuit Certification instead of Reachability).

2.2 Isolation

We now establish Theorem 6. The following proposition[8] shows that it suffices to construct a Turing machine that accepts Reachability unambiguously on layered digraphs in time $\mathrm{poly}(n)$ and space $O((\log n)^{3/2})$.

▶ **Proposition 10.** Reachability *on layered digraphs is hard for* NL *under logspace mapping reductions that preserve the number of solutions.*

Given our weight assignment generator $\Gamma^{(\mathrm{reach})}$, a natural approach towards computing Reachability unambiguously on a given layered instance (G, s, t) is to go over the list of all weight assignments w produced by $\Gamma^{(\mathrm{reach})}$, pick the first one that is min-isolating for G, and use it to decide the given instance (G, s, t). In fact, the earlier improvement from Reachability $\in \mathrm{R} \cdot \mathrm{PromiseUL}$ [30] to Reachability $\in \mathrm{R} \cdot (\mathrm{UL} \cap \mathrm{coUL})$ [51] can be viewed as following the same approach. Instead of the list of weight assignments obtained from $\Gamma^{(\mathrm{reach})}$ (which is guaranteed to contain a min-isolating one), [51] uses a list of $2n^2$ random weight assignments of bitlength $O(\log n)$ (which contains a min-isolating one with probability at least 50%). The following ingredients are essential to get the approach to work.

▶ **Lemma 11** ([51])**.** *There exist unambiguous nondeterministic machines* $\mathrm{WeightEval}^{(\mathrm{reach})}$ *and* $\mathrm{WeightCheck}^{(\mathrm{reach})}$ *such that for every digraph* $G = (V, E)$ *on* n *vertices, weight assignment* $w : V \mapsto \mathbb{N}$*, and* $s, t \in V$*:*

- $\mathrm{WeightCheck}^{(\mathrm{reach})}(G, w)$ *decides whether or not* w *is min-isolating for* G*, and*
- $\mathrm{WeightEval}^{(\mathrm{reach})}(G, w, s, t)$ *computes* $w(G, s, t)$ *provided* w *is min-isolating for* G*.*

Both machines run in time $\mathrm{poly}(\log(W), n)$ *and space* $O(\log(W) + \log(n))$*, where* W *denotes an upper bound on the finite values of* $w(G, u, v)$ *for* $u, v \in V$*.*

Note that the machine $\mathrm{WeightEval}^{(\mathrm{reach})}$ does not simply go over all integers μ from 0 to W and check whether a path from s to t of weight μ exists (knowing that it is unique if it exists) until the first success or the weight range is exhausted. That process would take at least W steps in the worst case, whereas the machine $\mathrm{WeightEval}^{(\mathrm{reach})}$ from Lemma 11 runs in time $\mathrm{poly}(\log(W), n)$.

Like [51], we call $\mathrm{WeightCheck}^{(\mathrm{reach})}(G, w)$ for each w from the list up and until the first success, and then call $\mathrm{WeightEval}^{(\mathrm{reach})}(G, w, s, t)$ with that first successful w. This describes a deterministic machine for Reachability on layered digraphs that makes calls to the unambiguous nondeterministic machines $\mathrm{WeightCheck}^{(\mathrm{reach})}$ and $\mathrm{WeightEval}^{(\mathrm{reach})}$. The result is an unambiguous nondeterministic machine assuming the following general convention regarding the behavior of a machine M making a call to a nondeterministic machine N: On any computation path on which N rejects, M halts and rejects; on any accepting computation path of N, M continues the path assuming the output of N as the result of the call.

Going over the list of all weight assignments produced by $\Gamma^{(\mathrm{reach})}$ is done by going over all seeds σ, and producing the required bits of $w = \Gamma^{(\mathrm{reach})}(\sigma)$ from σ on the fly whenever they are needed, without storing them. Given the logspace computability of

[8] The property that the mapping reductions preserve the number of solutions is not needed here but will be used later.

$\Gamma^{(\mathrm{reach})}$, the resulting unambiguous machine for Reachability on layered digraphs runs in time $2^R \cdot \mathrm{poly}(\log(W), n)$ and space $O(R + \log(W) + \log n)$, where R denotes the seed length of $\Gamma^{(\mathrm{reach})}$, and W the maximum path length under a weight assignment that $\Gamma^{(\mathrm{reach})}$ produces. With the parameters of $\Gamma^{(\mathrm{reach})}$ stated in Theorem 5 this gives time $n^{O(\sqrt{\log n})}$ and space $O((\log n)^{3/2})$.

In order to reduce the running time to $n^{O(1)}$ while keeping the space bound $O((\log n)^{3/2})$, we improve over the exhaustive search over all seeds of $\Gamma^{(\mathrm{reach})}$ by exploiting the internal structure of $\Gamma^{(\mathrm{reach})}$. Recall from the final construction in Section 2.1 that the seed σ consists of $\Delta = O(\sqrt{\log n})$ parts of $O(\log n)$ bits, each describing a hash function h_i from the family $\Gamma^{(\mathrm{hashing})}$ from Fact 9. The hash functions $h_1, \ldots, h_i$ define a weight assignment $w_{i \cdot \Delta}$ that is intended to have the following property: $w_{i \cdot \Delta}$ is min-isolating for each block of depth $2^{i \cdot \Delta}$ of G. We construct (the seeds σ_i for) the hash functions h_i one by one, maintaining the intended property as an invariant for $i = 0, 1, \ldots, \Delta$. The invariant trivially holds for $i = 0$. In the step from $i - 1$ to i for $i \in [\Delta]$, we go over all possible seeds σ_i for $\Gamma^{(\mathrm{hashing})}_{m,r}$, consider $h_i \doteq \Gamma^{(\mathrm{hashing})}_{m,r}(\sigma_i)$, check whether or not the weight assignment $w_{i \cdot \Delta}$ defined by the already determined $h_1, \ldots, h_{i-1}$ and the current choice for h_i maintains the invariant, and select the first σ_i for which it does. Each check is performed by running $\textsc{WeightCheck}^{(\mathrm{reach})}(B, w)$ for each of the blocks B of depth 2^i, passing if and only if all of them pass. The correctness argument from Section 2.1 guarantees that the search always succeeds. Once we arrive at $w = w_\Delta$, we run $\textsc{WeightEval}^{(\mathrm{reach})}(G, w, s, t)$ as before. Note that the number of choices for σ_i that need to be examined for each $i \in [\Delta]$ is $n^{O(1)}$. It follows that the resulting machine runs in time $n^{O(1)}$ and space $O((\log n)^{3/2})$, and unambiguously decides Reachability on layered digraphs.

This finishes the proof of Theorem 6. A more formal proof in the setting of Circuit Certification and LogCFL is given in Section 3.3.

3 Circuit Certification and LogCFL

We start this section with some background on Circuit Certification and LogCFL, including known isolation results. We then state and formally prove our positive results for this setting.

3.1 Background

Gal and Wigderson [30] applied their approach for isolating Reachability also to the following computational problem.

▶ **Definition 12** (Circuit certification). Circuit Certification denotes the computational problem that maps an input $x \doteq (C, z, g)$ composed of a Boolean circuit C, an input z for C, and a gate g of C, to the set of certificates for g in C on input z.

A *certificate* for a gate g in a Boolean circuit C on an input z is a minimal[9] subcircuit F of C with output gate g that accepts z, written $F(z) = 1$. Based on De Morgan's laws, one can always push the negations in a circuit to the inputs without changing the input/output behavior or the depth of the circuit, while at most doubling its size. On any given input z,

[9] The restriction of minimality is imposed in some references (e.g., [51]) but not in others (e.g., [30]). We impose it as it allows for a bijection between certificates and accepting computation paths in the machine characterization of LogCFL.

there is a simple bijection between the certificates for the transformed circuit and for the original one. Thus, it suffices to consider circuits where negations appear on the inputs only. In such a circuit C on input z, a certificate for a gate g satisfying $g(z) = 1$ can be constructed in the following recursive fashion, starting from the subcircuit of C rooted at g: If g is an AND gate, keep each incoming wire but replace its originating gate by a certificate for that gate. If g is an OR gate, keep a single incoming wire from a gate v satisfying $v(z) = 1$, and replace v by a certificate for v. If g is a leaf (necessarily evaluating to 1), keep it.

[30] assigns random weights w to the wires E of C. In order to facilitate the translation of the search for a certificate F for $x \doteq (C, z, g)$ of a given weight τ into an equivalent instance $f(x)$ of Circuit Certification, the certificate is conceptually first expanded into an equivalent formula in the standard way by duplicating gates, wires, and their weights. The weight of the certificate F is then defined as the weight of this formula seen as a weighted tree. Equivalently, along the lines of the above process for constructing a certificate, the weight of a certificate F for g can be defined recursively as the sum of the weights of the wires feeding into g and the weights of the certificates that F induces for their originating gates. Thus, the weight of a certificate is not merely the sum of the weights of the edges in the certificate, but a linear combination of those weights with nonnegative integer coefficients. The Isolation Lemma can be extended to this setting, namely to families of multisets over the universe E, and guarantees with probability at least $1 - 1/q$ that g has a unique certificate of minimum weight when $w : E \mapsto [q \cdot |E|]$ is chosen uniformly at random. The number of times a wire can appear in the multiset (the coefficient in the linear combination) can be as large as the maximum product of the fan-ins of the AND gates on a path in C from the inputs to g. As a consequence, only circuits of low depth in which the fan-in of the AND gates is small can be handled efficiently. More specifically, [30] considers *shallow semi-unbounded circuits*. "Shallow" means that the depth is bounded by $\log_2(n)$, where n denotes the number of gates. "Semi-unbounded" means that the fan-in of the AND gates is bounded by two (and that negations appear on the inputs only).

Shallow semi-unbounded circuits are intimately connected to the complexity class LogCFL of languages that reduce to a context-free language under logspace mapping reductions. The class can be defined equivalently as the languages accepted by logspace-uniform families of shallow semi-unbounded circuits of polynomial size, the non-uniform version of which is denoted as SAC^1 [61]. The class LogCFL can also be characterized as the languages accepted by nondeterministic machines that run in polynomial time and logarithmic space, and are equipped with an auxiliary stack that does not count towards the space bound [55]. Such machines are sometimes called auxiliary pushdown automata, and the class of languages accepted by such machines running in time t and space s is denoted as AuxPDA-TISP(t, s). The corresponding subclass for unambiguous machines is written as UAuxPDA-TISP(t, s). For any given problem in LogCFL and any input x, there is a logspace computable and logspace invertible bijection between the certificates for the circuits underlying the logspace-uniform SAC^1 characterization, and the accepting computation paths of the machine underlying the AuxPDA-TISP$(\mathrm{poly}(n), O(\log n))$ characterization. It follows that the restriction of Circuit Certification to shallow semi-unbounded circuits is complete for LogCFL under logspace mapping reductions, and that logspace computable and recoverable disambiguations for that problem and for the entire class are equivalent.

Gal and Wigderson obtained a randomized disambiguation for Circuit Certification on shallow semi-bounded circuits that has success probability $1/\mathrm{poly}(n)$, is computable in logspace with two-way access to the random bits, and is recoverable in logspace. This implies that LogCFL $\subseteq \mathrm{R} \cdot \mathrm{Promise}\mathcal{C}$ where $\mathcal{C} \doteq$ UAuxPDA-TISP$(\mathrm{poly}(n), O(\log n))$. Reinhardt and

Allender [51] strengthened this result to $\mathrm{LogCFL} \subseteq \mathrm{R} \cdot (\mathcal{C} \cap \mathrm{co}\mathcal{C})$, replacing the condition on the weight assignment w by the requirement that w is min-isolating for *every* gate of C on input z (not just the specified gate g). This implies that $\mathrm{LogCFL} \subseteq (\mathcal{C} \cap \mathrm{co}\mathcal{C})/\mathrm{poly}$ and that a disambiguation for Circuit Certification on shallow semi-unbounded circuits can be computed in logspace with polynomial advice.

3.2 Weight Assignment Generator

Analogous to the setting of Reachability and NL, our isolations for Circuit Certification and LogCFL hinge on an efficient min-isolating weight assignment generator. Although not essential, it is more convenient for us to assign weights to the *gates* rather than the wires.

Let us formally define what min-isolation means in the context of Circuit Certification. We view a Boolean circuit C as an acyclic digraph $C = (V, E)$, where V represents the gates of the circuit, and E the wires. Each leaf (vertex of indegree zero) is labeled with a literal (input variable or its negation) or a Boolean constant (0 or 1); each other vertex is labeled with AND or OR. We consider circuits with and without a single designated output gate.

▶ **Definition 13** (Min-isolating weight assignment for Circuit Certification)**.** Let $C = (V, E)$ be a circuit. A weight assignment for C is a mapping $w : V \mapsto \mathbb{N}$. The weight $w(F)$ of a certificate F with output v equals $w(v)$ plus the sum over all gates u that feed into v in F, of the weight of the certificate with output u induced by F. For an input z for C, and $g \in V$, $w(C, z, g)$ denotes the minimum of $w(F)$ over all certificates F for (C, z, g), or ∞ if no certificate exists. The weight assignment w is min-isolating for (C, z, g) if there is at most one certificate F for (C, z, g) with $w(F) = w(C, z, g)$. For $U \subseteq V$, w is min-isolating for (C, z, U) if w is min-isolating for (C, z, u) for each $u \in U$. We call w min-isolating for (C, z) if w is min-isolating for (C, z, V).

Note that the weight $w(F)$ of a certificate F for a gate g is a linear combination of the weights $w(v)$ for $v \in V$ with coefficients that are natural numbers. The sum of the coefficients in any given layer below g is at most 2^ℓ, where ℓ denotes the number of AND layers between that layer and g (inclusive).

We restrict attention to semi-unbounded circuits that are *layered* and *alternating*. A circuit is layered if the underlying digraph is layered and all leaves appear in the same layer. A circuit is alternating if on every path the non-leaves alternate between AND and OR. More formally, for a circuit $C = (V, E)$ of depth d with n gates we have that $V = \cup_{i \in [\![d]\!]} V_i$ where $V_i \subseteq [n] \times \{i\}$ and $E \subseteq \cup_{i \in [d]} (V_{i-1} \times V_i)$. Vertices in V_0 are labeled with literals and constants only. Every other layer V_i contains only AND gates or only OR gates, depending on the parity of i.

With the above conventions we can view weight assignments to the gates as mappings $w : [n] \times [\![d]\!] \mapsto \mathbb{N}$. We construct such assignments inside the following weight assignment generator $\Gamma^{(\mathrm{cert})} = (\Gamma^{(\mathrm{cert})}_{n,d})_{n,d \in \mathbb{N}}$, which is indexed by the number of gates n and the depth d. The domain of the weight assignments given by $\Gamma^{(\mathrm{cert})}_{n,d}$ is $D_{n,d} \doteq [n] \times [\![d]\!]$, enabling the weight assignment of a gate to depend on the layer the gate belongs to.

▶ **Theorem 14.** *There exists a weight assignment generator $\Gamma^{(\mathrm{cert})} = (\Gamma^{(\mathrm{cert})}_{n,d})_{n,d \in \mathbb{N}}$ that is computable in space $O(\log n)$ and has seed length and bitlength $O(\sqrt{d} \log n)$ such that for every layered alternating semi-unbounded Boolean circuit C of depth d with n gates and any input z for C,*

$$\Pr_w[w \text{ is min-isolating for } (C, z)] \geq 1 - 1/n,$$

where w is chosen uniformly at random from $\Gamma^{(\mathrm{cert})}_{n,d}$.

The essential ingredient in the proof of Theorem 14 is the following formalization of the combined approach from Section 2.1 for the setting of Circuit Certification. It turns a weight assignment that is min-isolating for all gates up to some layer into one that is min-isolating for all gates up to some higher layer, and only assigns new weights to the AND gates of the layers in between. For ease of notation, we assume that the depth is even (say $d = 2\ell$ for some $\ell \in \mathbb{N}$), that we jump from an even layer $2k$ to some higher even layer $2k'$, and that the layer V_1 next to the leaves consists of ANDs. Thus, odd layers consist of AND gates, and positive even layers of OR gates.

▶ **Lemma 15.** *There exists a weight assignment generator $\Gamma^{(\mathrm{cert,step})} = (\Gamma^{(\mathrm{cert,step})}_{n,\ell,k,k'})$ for $n, \ell, k, k' \in \mathbb{N}$ with $k \le k' \le \ell$ and domain $D_{n,\ell,k,k'} \doteq [n] \times [\![2\ell]\!]$ that is computable in space $O(\log n)$, has seed length $O(\log n)$ and bitlength $O((k'-k)\log n)$, and has the following property for every layered alternating semi-unbounded Boolean circuit $C = (V, E)$ of depth $d \doteq 2\ell$ with n gates and layers $V_0, V_1, \ldots, V_d$ where layer V_1 consists of AND gates, and for every input z for C: If $w : V \mapsto \mathbb{N}$ is a weight assignment that is min-isolating for $(C, z, V_{\le 2k})$, where $V_{\le i} \doteq \cup_{j \le i} V_j$, then*

$$\Pr_{\sigma}[w + \Gamma^{(\mathrm{cert,step})}_{n,\ell,k,k'}(\sigma) \text{ is min-isolating for } (C, z, V_{\le 2k'})] \ge 1 - 1/n^2,$$

where the seed σ is chosen uniformly at random. Moreover, $\Gamma^{(\mathrm{cert,step})}_{n,\ell,k,k'}(\sigma)$ assigns nonzero weights only to $\cup_{j \in [k+1,k']} L_j$, where $L_j \doteq V_{2j-1}$ denotes the jth AND layer.

Proof. Let C be a circuit as in the statement of the lemma, z an input for C, and $w : V \mapsto \mathbb{N}$ a weight assignment that is min-isolating for $(C, z, V_{\le 2k})$.

Pick $h : D \mapsto [r]$ with $D = D_{n,\ell,k,k'} \doteq [n] \times [\![2\ell]\!]$ uniformly at random from $\Gamma^{(\mathrm{hashing})}_{n(2\ell+1),r}$, identifying $[n(2\ell+1)]$ and $[n] \times [\![2\ell]\!]$ in a natural way. For a given h, we define a sequence of weight assignments $w_j : V \mapsto \mathbb{N}$ for $j = k, k+1, \ldots, k'$ as follows: $w_k = w$, and for $i \in [k'-k]$ and $g \in V$:

$$w_{k+i}(g) = \begin{cases} w_{k+i-1}(g) + h(g) \cdot b^{i-1} & \text{if } g \in L_{k+i} \\ w_{k+i-1}(g) & \text{otherwise,} \end{cases}$$

where b is a positive integer to be determined.

For $g \in V$, we denote by $\mu_j(g) \doteq w_j(C, z, g)$ the minimum weight of a certificate for (C, z, g) with respect to w_j, or ∞ if no certificate exists. We show that if b and r are sufficiently large polynomials in n, then with probability at least $1 - 1/n^2$ the following invariant holds for $i \in [\![k'-k]\!]$:

$$w_{k+i} \text{ is min-isolating for } (C, z, V_{\le 2(k+i)}). \tag{8}$$

We make the following observations:

- By the hypothesis on w the invariant holds for $i = 0$.
- For $i \in [k'-k]$, the invariant for $i-1$ implies that w_{k+i} is min-isolating for $(C, z, V_{\le 2(k+i-1)})$. The reason is that for gates $g \in V_{\le 2(k+i-1)}$, whether a weight assignment is min-isolating for (C, z, g) only depends on the weights of the gates in $V_{\le 2(k+i-1)}$. As w_{k+i-1} and w_{k+i} agree on that set, the invariant for $i-1$ implies that w_{k+i} is min-isolating for (C, z, g).
- For $i \in [k'-k]$, the invariant for $i-1$ implies that w_{k+i} is min-isolating for $(C, z, V_{2(k+i)-1})$. This follows because $V_{2(k+i)-1}$ is an AND layer. A certificate for an AND gate $g \in V_{2(k+i)-1}$ is the AND of certificates for gates $u, v \in V_{2(k+i-1)}$ feeding into g, and $w_{k+i}(C, z, g) = w_{k+i}(g) + w_{k+i}(C, z, u) + w_{k+i}(C, z, v)$. Since w_{k+i} and w_{k+i-1} agree on $V_{2(k+i-1)}$, the invariant for $i-1$ implies that w_{k+i} is min-isolating for (C, z, g).

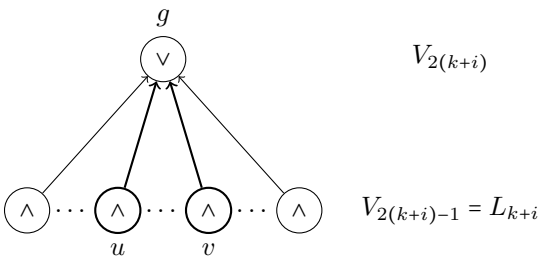


Figure 2

Thus, in order to show that the invariant is maintained from $i-1$ to i for $i \in [k'-k]$, it suffices to show that w_{k+i} is min-isolating for $(C, z, V_{2(k+i)})$ assuming the invariant holds for $i-1$. The following claim provides a sufficient condition.

▶ **Claim 16.** *Let $i \in [k'-k]$, and $g \in V_{2(k+i)}$ with $g(z) = 1$. Suppose that $b \geq 4r$ and that w_{k+i-1} is min-isolating for $(C, z, V_{\leq 2(k+i-1)})$. If for all distinct $u, v \in L_{k+i} \doteq V_{2(k+i)-1}$ that feed into g*

$$\left\lfloor \frac{\mu_k(u)}{b^{i-1}} \right\rfloor + h(u) \notin \left\lfloor \frac{\mu_k(v)}{b^{i-1}} \right\rfloor + h(v) + \{-1, 0, 1\}, \tag{9}$$

then w_{k+i} is min-isolating for (C, z, g).

See Figure 2 for an illustration.

Proof of Claim 16. Since g is an OR gate, a certificate F_g for (C, z, g) consists of an edge from g to one of its inputs v for which $v(z) = 1$, and a certificate F_v for v. As $w_{k+i}(F_v) = w_{k+i-1}(F_v) + h(v) \cdot b^{i-1}$, it follows that the min-weight certificates for v under w_{k+i-1} and under w_{k+i} are the same. Thus, v has a unique min-weight certificate under w_{k+i},

$$\mu_{k+i}(v) = \mu_{k+i-1}(v) + h(v) \cdot b^{i-1}, \tag{10}$$

and the following condition is sufficient to guarantee that w_{k+i} is min-isolating for (C, z, g): For all distinct inputs $u, v \in L_{k+i} \doteq V_{2(k+i)-1}$ that feed into g

$$\mu_{k+i}(u) \neq \mu_{k+i}(v). \tag{11}$$

We argue that (11) follows from (9) as long as $b \geq 4r$.

For $v \in L_{k+i}$ with $v(z) = 1$, let F_v denote a min-weight certificate for v under w_k. We have that

$$\mu_k(v) \leq \mu_{k+i-1}(v) \leq w_{k+i-1}(F_v) \leq w_k(F_v) + 4r \cdot b^{i-2} = \mu_k(v) + 4r \cdot b^{i-2}. \tag{12}$$

The first inequality follows because $w_{k+i-1} \geq w_k$, and the second one and the last one from the definition of μ. For the third inequality, note that w_{k+i-1} is obtained from w_k by adding weights to the vertices in the AND layers below L_{k+i}. In particular, for a $u \in L_{k+i-j}$ we have that $w_{k+i-1}(u) = w_k(u) + h(v) \cdot b^{i-1-j} \leq w_k(u) + r \cdot b^{i-1-j}$. The sum of the coefficients that the weights of the vertices in L_{k+i-j} receive in $w_{k+i-1}(F_v)$ is at most 2^j. Summing over all such layers with $j > 0$ we have that

$$w_{k+i-1}(F_v) \leq w_k(F_v) + r \cdot \sum_{j=1}^{i-1} 2^j b^{i-1-j} \leq w_k(F_v) + 4r \cdot b^{i-2}$$

for $b \geq 4$.

After division by b^{i-1}, (12) shows that

$$\frac{\mu_k(v)}{b^{i-1}} \leq \frac{\mu_{k+i-1}(v)}{b^{i-1}} \leq \frac{\mu_k(v)}{b^{i-1}} + \frac{4r}{b},$$

which implies that

$$\left\lfloor \frac{\mu_k(v)}{b^{i-1}} \right\rfloor \leq \left\lfloor \frac{\mu_{k+i-1}(v)}{b^{i-1}} \right\rfloor \leq \left\lfloor \frac{\mu_k(v)}{b^{i-1}} \right\rfloor + 1 \tag{13}$$

for $b \geq 4r$. In combination with the hypothesis (9), (13) implies that

$$\left\lfloor \frac{\mu_{k+i-1}(u)}{b^{i-1}} \right\rfloor + h(u) \neq \left\lfloor \frac{\mu_{k+i-1}(v)}{b^{i-1}} \right\rfloor + h(v),$$

which by (10) in turn implies (11) after undoing the division. This finishes the proof of Claim 16. ◀

Each individual disambiguation requirement (9) can be written as three conditions of the form (4). By Fact 9, each of these three conditions individually holds with probability at least $1 - 1/r$. There are at most n^3 disambiguation requirements over all $i \in [k' - k]$, namely n choices for each of g, u, and v. A union bound shows that they all hold simultaneously with probability at least $1 - 3n^3/r$, which is at least $1 - 1/n^2$ for $r \geq 3n^5$. Whenever they hold, we know that the invariant (8) holds for each $i \in [\![k' - k]\!]$, and in particular that $w_{k'}$ is min-isolating for $(C, z, V_{\leq 2k'})$.

This leads to the following definition of $\Gamma^{(\mathrm{cert,step})}$: $\Gamma^{(\mathrm{cert,step})}_{n,\ell,k,k'}$ takes a seed σ for $\Gamma^{(\mathrm{hashing})}_{n(2\ell+1),r}$, considers $h = \Gamma^{(\mathrm{hashing})}_{n(2\ell+1),r}(\sigma)$ as a function $h : D \mapsto [r]$ with $D = D_{n,\ell,k,k'} \doteq [n] \times [\![2\ell]\!]$, and for $g \in [n] \times \{j\}$ sets

$$(\Gamma^{(\mathrm{cert,step})}_{n,\ell,k,k'}(\sigma))(g) = \begin{cases} h(g) \cdot b^{(j-1)/2-k} & \text{for odd } j \in [2k+1, 2k'-1] \\ 0 & \text{otherwise.} \end{cases}$$

The above analysis shows that $\Gamma^{(\mathrm{cert,step})}$ has the required min-isolating property. By setting b to the first power of 2 that is at least $4r$ with $r = 3n^5$, the bitlength becomes $O((k'-k)\log b)) = O((k'-k)\log n)$. The other required properties follow from the properties of the universal family $\Gamma^{(\mathrm{hashing})}_{m,r}$ given in Fact 9. They imply that $\Gamma^{(\mathrm{cert,step})}_{n,\ell,k.k'}$ has seed length $O(\log(|D_{n,\ell,k,k'}| \cdot r) = O(\log n)$. As each bit of $(\Gamma^{(\mathrm{cert,step})}(\sigma))(g)$ equals an easily determined bit of $h(g)$, the logspace computability of the universal family of hash functions implies the logspace computability of $\Gamma^{(\mathrm{cert,step})}_{n,\ell,k,k'}$. This completes the proof of Lemma 15. ◀

We now turn to the proof of the theorem.

Proof of Theorem 14. Let C be a circuit as in the statement of the theorem with layers $V_j \subseteq [n] \times \{j\}$ for $j \in [\![d]\!]$, and let z be an input for C. Consider first the case where the layer V_1 of C next to the leaves consists of ANDs.[10]

If d is even and of the form $d = 2\ell$ with $\ell = \Delta^2$ for some $\Delta \in \mathbb{N}$, we can apply Lemma 15 Δ times successively, starting from an arbitrary weight assignment w_0. The ith application sets $k = k_i \doteq (i-1) \cdot \Delta$ and $k' = k'_i \doteq i \cdot \Delta$, uses a fresh seed σ_i for $\Gamma^{(\mathrm{cert,step})}_{n,\ell,k_i,k'_i}$, sets $w_{i \cdot \Delta} =$

[10] This is the only case we need for the proof of Theorem 17.

$w_{(i-1)\cdot\Delta} + \Gamma^{(\mathrm{cert,step})}_{n,\ell,k_i,k'_i}(\sigma_i)$, and tries to maintain the invariant that $w_{i\cdot\Delta}$ is min-isolating for $(C, z, V_{\leq 2i\cdot\Delta})$. We end up with $w_\ell = w_0 + \Gamma^{(\mathrm{cert,odd})}_{n,d}(\sigma_1, \sigma_2, \ldots, \sigma_\Delta)$, where

$$\Gamma^{(\mathrm{cert,odd})}_{n,d}(\sigma_1, \sigma_2, \ldots, \sigma_\Delta) \doteq \sum_{i\in[\Delta]} \Gamma^{(\mathrm{cert,step})}_{n,\ell,k_i,k'_i}(\sigma_i). \tag{14}$$

The superscript "odd" in $\Gamma^{(\mathrm{cert,odd})}$ refers to the fact that only odd layers receive nonzero values under weight assignments generated by $\Gamma^{(\mathrm{cert,odd})}$. The probability that the ith application breaks the invariant is at most $1/n^2$. By a union bound, the probability that the invariant fails at the end is at most $\Delta/n^2 \leq 1/n$. Thus, for any fixed $w_0 : [n] \times [\![d]\!] \mapsto \mathbb{N}$, $w_0 + \Gamma^{(\mathrm{cert,odd})}_{n,d}$ is min-isolating for (C, z) with probability at least $1 - 1/n$. The seed length of $\Gamma^{(\mathrm{cert,odd})}_{n,d}$ is Δ times the one of $\Gamma^{(\mathrm{cert,step})}_{n,d,\cdot,\cdot}$, i.e., $O(\Delta \log n) = O(\sqrt{d} \log n)$. The maximum weight assigned by $\Gamma^{(\mathrm{cert,odd})}_{n,d}$ is at most Δ times the one assigned by $\Gamma^{(\mathrm{cert,step})}_{n,d,\cdot,\cdot}$, so the bitlength of $\Gamma^{(\mathrm{cert,odd})}_{n,d}$ is $O(\log(\Delta) + \Delta \cdot \log n) = O(\sqrt{d} \log n)$. The logspace computability of $\Gamma^{(\mathrm{cert,step})}$ and the fact that iterated addition can be computed in logspace (see, e.g., [62] imply that $\Gamma^{(\mathrm{cert,odd})}_{n,d}$ is computable in space $O(\log n)$.

Other values of d can be handled by conceptually extending the circuit with successive matchings until the depth is of the form $2\Delta^2$, applying the above construction, and then only using the part needed. As the smallest such Δ still satisfies $\Delta = \Theta(\sqrt{d})$, the parameters remain the same up to constant factors. Thus, we have a weight assignment generator $\Gamma^{(\mathrm{cert,odd})}$ with all the properties required of $\Gamma^{(\mathrm{cert})}$ in the case where the layer V_1 of C consists of ANDs.

To handle the case where V_1 consists of ORs, we can conceptually split every wire (u, v) from a leaf u to $v \in V_1$ into two by inserting a fresh AND gate g and replacing (u, v) by (u, g) and (g, v). We then apply the construction for the case where V_1 consists of ANDs, and finally undo the splitting again, transfering the weight of each fresh AND gate g to the leaf u that feeds into it. This results in a weight assignment generator $\Gamma^{(\mathrm{cert,even})}$ that only assigns nonzero weights to the even layers, and has all the properties required of $\Gamma^{(\mathrm{cert})}$ for circuits C where the layer V_1 next to the leaves consists of ORs. For any such circuit C, input z for C, and any fixed $w'_0 : [n] \times [\![d]\!] \mapsto \mathbb{N}$, we have that $w'_0 + \Gamma^{(\mathrm{cert,even})}_{n,d}$ is min-isolating for (C, z) with probability at least $1 - 1/n$.

We claim that

$$\Gamma^{(\mathrm{cert})}_{n,d} \doteq \Gamma^{(\mathrm{cert,odd})}_{n,d} + \Gamma^{(\mathrm{cert,even})}_{n,d}$$

satisfies all requirements irrespective of the type of V_1, provided that we pick the seeds for $\Gamma^{(\mathrm{cert,odd})}_{n,d}$ and $\Gamma^{(\mathrm{cert,even})}_{n,d}$ independently. This follows from the above analysis by setting $w_0 = \Gamma^{(\mathrm{cert,even})}_{n,d}$ and $w'_0 = \Gamma^{(\mathrm{cert,odd})}_{n,d}$, and finishes the proof of Theorem 14. ◀

3.3 Isolation

We use the weight assignment generator from Theorem 14 to establish the following result

▶ **Theorem 17.** $\mathrm{LogCFL} \subseteq \mathrm{UAuxPDA\text{-}TISP}(\mathrm{poly}(n), (\log n)^{3/2})$.

In words: Every language in the class LogCFL can be accepted by a nondeterministic machine equipped with a stack that does not count towards the space bound, that runs in polynomial time and $O((\log n)^{3/2})$ space, and has at most one accepting computation path on every input.

By the following proposition, it suffices to construct such a machine for Circuit Certification on shallow layered alternating semi-unbounded circuits.

▶ **Proposition 18.** Circuit Certification *on shallow layered alternating semi-unbounded Boolean circuits is hard for* LogCFL *under logspace mapping reductions that preserve the number of solutions.*

Our unambiguous machine for Circuit Certification hinges on our weight assignment generator for the problem as well as the following unambiguous machines.

▶ **Lemma 19.** *There exist unambiguous nondeterministic machines* $\text{WeightCheck}^{(\text{cert})}$ *and* $\text{WeightEval}^{(\text{cert})}$*, each equipped with a stack that does not count towards the space bound, such that for every layered semi-unbounded Boolean circuit* $C = (V, E)$ *of depth* d *with* n *gates, every input* z *for* C*, weight assignment* $w : V \mapsto \mathbb{N}$*, and* $g \in V$*:*

- $\text{WeightCheck}^{(\text{cert})}(C, z, w)$ *decides whether or not* w *is min-isolating for* (C, z)*, and*
- $\text{WeightEval}^{(\text{cert})}(C, z, w, g)$ *computes* $w(C, z, g)$ *provided* w *is min-isolating for* (C, z)*.*

Both machines run in time $\text{poly}(2^d, \log(W), n)$ *and space* $O(d + \log(W) + \log(n))$*, where* W *denotes an upper bound on the finite values* $w(C, z, g)$ *for* $g \in V$*.*

Lemma 19 is an improvement of a result in [51] that follows along the same lines but has a better dependency of the running time on W, namely polynomial in $\log(W)$ instead of polynomial in W. As our weight assignment generator yields values of $W = n^{\Theta(\sqrt{\log n})}$, the improvement is necessary to make sure that our unambiguous machine for Circuit Certification on shallow layered alternating semi-unbounded circuits run in polynomial time.

We now have all the ingredients to establish our efficient unambiguous machines for LogCFL.

Proof of Theorem 17. By way of Proposition 18, it suffices to construct an unambiguous machine that decides[11] Circuit Certification on layered alternating semi-unbounded Boolean circuits $C = (V, E)$ of size n and depth $d \le \log(n)$, and runs in time $n^{O(1)}$ and space $O((\log n)^{3/2})$ when equipped with a stack that does not count towards the space bound. In fact, thanks to simple manipulations described earlier, it suffices to consider the case where the depth d is of the form $d = 2\Delta^2$ for $\Delta \in \mathbb{N}$, and where the layer next to the leaves consists of ANDs. We claim that the machine CircuitEval described in Algorithm 1 does the job.

Consider the version of our weight assignment generator $\Gamma^{(\text{cert})}$ from Theorem 14 that is geared towards such circuits, namely $\Gamma^{(\text{cert,odd})}$ given by (14). We know that on most seeds $\Gamma^{(\text{cert,odd})}_{n,d}$ produces a weight assgnment w that is min-isolating for (C, z). The machine CircuitEval in Algorithm 1 constructs such a seed. In fact, it constructs the lexicographically first such seed.

Recall that the seed σ consists of Δ parts $\sigma_i \in \{0,1\}^{s(n,d)}$ for $i \in [\Delta]$, where $s(n, d)$ denotes the seed length of $\Gamma^{(\text{cert,step})}_{n,d,\cdot,\cdot}$. Note that $w \doteq \Gamma^{(\text{cert,odd})}_{n,d}(\sigma_1, \ldots, \sigma_\Delta)$ is min-isolating for (C, z) if and only if

$$w_{i\cdot\Delta} \doteq \sum_{j=1}^{i} \Gamma^{(\text{cert,step})}_{n,d,(j-1)\Delta,j\Delta}(\sigma_j) \text{ is min-isolating for } (C, V_{\le 2i\Delta}) \tag{15}$$

for each $i \in [\Delta]$. This enables a prefix search for the lexicographically first σ for which w is min-isolating for (C, z). The first part of Algorithm 1 implements this search. In the ith iteration it finds the lexicographically first σ_i satisfying the invariant (15), given values for

[11] In fact, we only need to construct a machine that *accepts* the language, but we naturally get the stronger notion of one that *decides* the language.

Algorithm 1: CircuitEval(C, z, g)

Input : $C = (V, E)$: layered semi-unbounded circuits of depth d with layers $V_0, V_1, \ldots, V_d$
z: input for C
$g \in V$
Promise : $d = 2\Delta^2$ for $\Delta \in \mathbb{N}$ and V_1 consists of ANDs
Output : $g(z)$

```
1  for i ← 1 to Δ do
2      foreach σ_i ∈ {0,1}^{s(n,d)} in lex order do
3          isolating ← true;
4          foreach v ∈ V_{≤2iΔ} in lex order do
5              if not WeightCheck^(cert)(C_v, z, Σ_{j=1}^{i} Γ^(cert,step)_{n,d,(j−1)·Δ,j·Δ}(σ_j)) then
6                  isolating ← false;
7                  exit the for loop over v;
8          end
9          if isolating then exit the loop over σ_i;
10     end
11 end
12 if WeightEval^(cert)(C, z, Σ_{j=1}^{Δ} Γ^(cert,step)_{n,d,(j−1)·Δ,j·Δ}(σ_j), g) < ∞ then
13     accept and return 1
14 else accept and return 0;
```

$\sigma_1, \ldots, \sigma_{i-1}$ from prior iterations. In order to check whether a given candidate σ_i works, it runs the machine WeightCheck$^{(\text{cert})}(C_v, z, w_{i \cdot \Delta})$ for each $v \in V_{\leq 2i\Delta}$, where C_v denotes the subcircuit of C rooted at v.

Once σ is determined, CircuitEval calls WeightEval$^{(\text{cert})}(C, z, w, g)$ to compute $w(C, z, g)$, which is finite if and only if $g(z) = 1$.

The correctness of CircuitEval follows from maintaining the invariant (15) and the specifications of WeightCheck$^{(\text{cert})}$ and WeightEval$^{(\text{cert})}$. The unambiguity of CircuitEval follows from the unambiguity of WeightCheck$^{(\text{cert})}$ and WeightEval$^{(\text{cert})}$ (and the usual conventions regarding composing unambiguous machines).

We end with a time and space analysis of CircuitEval. Each run of line 5 takes time $\text{poly}(2^d, \log(W), n)$ and space $O(d + \log(W) + \log(n))$, where W is a bound on the path weights under w. This follows from the complexities of $\Gamma^{(\text{cert,odd})}$ and WeightCheck$^{(\text{cert})}$, and the fact that iterated addition is in logspace (see, e.g., [62]). The three loops add a multiplicative term of $\Delta \cdot 2^{s(n,d)} \cdot n$ to the running time, and an additive term of $\log(\Delta) + s(n, d) + \log(W)$ to the space bound. The time and space needed for the call to WeightEval$^{(\text{cert})}$ at the end is dominated by the rest of the computation. Since $\Delta = \Theta(\sqrt{d}) \leq \sqrt{\log n}$, $s(n, d) = O(\log n)$, and $W = 2^{O(\Delta \cdot \log(n))}$, the overall running time is $\text{poly}(2^d, n)$ and the space is $O(\sqrt{d} \log(n))$. This yields the stated complexities in the case of shallow circuits, for which $d \leq n$. ◀

4 Limitations

In this section we prove our "negative result" for isolating Reachability (Theorem 7) and a corresponding result for Circuit Certification.

Recall that we view a computational problem as a mapping $\Pi : X \mapsto 2^Y$, where $\Pi(x)$ for $x \in X$ represents the set of solutions on input x. One can also think of Π as defining a

relation $\pi : X \times Y \mapsto \{0,1\}$, where $\pi(x,y)$ indicates whether $y \in \Pi(x)$. We use the notation $L(\Pi)$ to denote the set (language) of instances $x \in X$ for which $\Pi(x) \neq \varnothing$.

The first part of Theorem 7 follows by verifying that the main result of Dell, Kabanets, Van Melkebeek, and Watanable [26] carries over to the space-bounded setting: If Π has an efficient pruning and π is efficiently computable, then $L(\Pi)$ can be decided efficiently. The prunings in this statement are deterministic or, more generally, randomized with probability of success at least $\frac{2}{3} + \frac{1}{\mathrm{poly}(n)}$. [26] showed that the statement holds when "efficient" means polynomial-time for any Π that satisfies certain additional properties, which all the classical problems like Satisfiability do. We observe that the argument in [26] also works when "efficient" means logspace, and that both Reachability and Circuit Certification have the required additional properties. This yields the first part of Theorem 7 and its counterpart for Circuit Certification. The second part follows from a slight modification of the argument.

The proof in [26] relies on a proposition of Ko's [41].

▶ **Proposition 20** ([41]). *Suppose that there exists a predicate $T : D \times D \mapsto \{0,1\}$ for some $D \subseteq X$ with the following properties:*

$$(\forall x, z \in D \cap L(\Pi))\; T(x,z) \vee T(z,x) \tag{16}$$

$$(\forall x, z \in D)\; z \in L(\Pi) \wedge T(z,x) \Rightarrow x \in L(\Pi) \tag{17}$$

Then for some $\ell \in [\![\lceil \log(|D|+1) \rceil]\!]$ there exists a sequence $z_1^*, \ldots, z_\ell^* \in D \cap \Pi$ such that for every $x \in D$

$$x \in L(\Pi) \Leftrightarrow (\exists i \in [\ell])\, T(z_i^*, x). \tag{18}$$

If the $\vee$ in (16) were replaced by an exclusive or, T would be a tournament, where $T(z,x)$ (an edge from z to x) means that x wins the duel between z and x. Equation (16) requires the digraph T to contain a tournament (and have a selfloop at every vertex), so every duel has at least one winner and can have two. Equation (17) can be interpreted as saying that winners of duels are more likely be to in $L(\Pi)$ in the following sense: If at least one of x or z is in $L(\Pi)$, then any winner of the duel between x and z is.

Proposition 20 follows from the fact that a tournament on $D \cap \Pi$ has a dominating set of logarithmic size. In the case where D represents all instances of a given size n (of which there are at most 2^n), Proposition 20 shows us via (18) how to decide $L(\Pi)$ efficiently on D with the help of the predicate T and the $\ell \cdot n \leq n^2$ bits of advice z_i^* for $i \in [\ell]$.

[26] constructs a (sufficiently) efficient predicate T satisfying (16) and (17) assuming the existence of an efficient deterministic pruning f for Π, that π is efficiently computable, and that Π allows an efficient disjoint union operator.

▶ **Definition 21** (Disjoint union of computational problems). Let $\Pi : X \mapsto 2^Y$ be a computational problem. A disjoint union operator for Π consists of a mapping $\sqcup : X \times X \mapsto X$ and a mapping $\tau : X \times X \times [2] \times Y \mapsto Y$ such that for all $x_1, x_2 \in X$, $|\Pi(x_1 \sqcup x_2)| = |\Pi(x_1)| + |\Pi(x_2)|$ and $\Pi(x_1 \sqcup x_2) = \cup_{i \in [2]} \tau(x_1, x_2, i, \Pi(x_i))$, where $\tau(x_1, x_2, i, W) \doteq \cup_{y \in W} \{\tau(x_1, x_2, i, y)\}$ for any $W \subseteq Y$.

$\sqcup$ maps a pair of instances (x_1, x_2) to an instance $x_1 \sqcup x_2$ whose solutions can be viewed as the disjoint union of the solutions of x_1 and of x_2, where $\tau(x_1, x_2, i, y_i)$ describes the translation of the solution $y_i \in \Pi(x_i)$ into the corresponding solution in $\Pi(x_1 \sqcup x_2)$.

Several of the classical computational problems Π allow simple disjoint union operators that are computable in logspace, meaning that both $\sqcup$ and τ in Definition 21 are computable

in logspace. Often times the underlying predicate π is computable in logspace as well. This is the case, among others, for Satisfiability, Reachability, and Circuit Certification.

▶ **Proposition 22.** Reachability *and* Circuit Certification *on shallow semi-unbounded circuits have disjoint union operators as well as underlying predicates that are computable in logspace. The same holds for their restrictions to layered digraphs, and to layered alternating circuits, respectively.*

The key insight in [26] is (i) that a pruning f applied to the disjoint union $x_1 \sqcup x_2$ implicitly selects an instance among x_1 and x_2 that is more likely to be positive in the above sense, and (ii) that the corresponding predicate T satisfying Ko's requirements (16) and (17) can be decided sufficiently efficiently on instances with few solutions provided that f is efficiently computable and that Π allows an efficient disjoint union operator. The corresponding predicate T can formally be defined as follows on all pairs of instances $(z, x) \in X \times X$:

$$T(z,x) \Leftrightarrow \begin{cases} \tau(z,x,1,\Pi(z)) \cap \Pi(f(z \sqcup x)) = \varnothing & \text{for } z \leq_{lex} x \\ \tau(x,z,2,\Pi(z)) \cap \Pi(f(x \sqcup z)) = \varnothing & \text{for } x \leq_{lex} z, \end{cases}$$

where $\leq_{lex}$ denotes the lexicographic ordering. The isolation property $|\Pi(f(\cdot))| \leq 1$ implies condition (16). The pruning property $\Pi(f(\cdot)) \subseteq \Pi(\cdot)$ implies condition (17). In the case of an instance z^* with a unique solution, say $\Pi(z^*) = \{y^*\}$, we can evaluate $T(z^*, z)$ as

$$T(z^*,x) \Leftrightarrow \begin{cases} \neg\pi(f(z^* \sqcup x), \tau(z^*,x,1,y^*)) & \text{for } z^* \leq_{lex} x \\ \neg\pi(f(x \sqcup z^*), \tau(x,z^*,2,y^*)) & \text{for } x \leq_{lex} z^*. \end{cases} \tag{19}$$

Given x, z^*, and y^*, the latter expression can be computed efficiently when all of π, f, $\sqcup$, and τ can. This leads to an efficient algorithm with advice for deciding $L(\Pi)$ on the instances of size n with at most one solution, where the advice consists of the strings (z_i^*, y_i^*) for $i \in [\ell]$. In order to decide $L(\Pi)$ on *any* instance $x \in X$, we first apply the pruning f, and then run the algorithm for instances with at most one solution on $f(x)$. This results in an efficient algorithm with polynomial advice for deciding $L(\Pi)$.

The above argument works for polynomial-time efficiency as well as for logspace efficiency. The polynomial-time incarnation yields the main result of [26] regarding the existence of deterministic polynomial-time prunings for Satisfiability. The logspace incarnation yields the first part of Theorem 7 regarding the existence of deterministic logspace prunings for Reachability as well as a corresponding result for Circuit Certification.

As for the second part of Theorem 7 and its counterpart for Circuit Certification, a min-isolating weight assignment $\omega(x, y)$ applied to the disjoint union $x_1 \sqcup x_2$ selects between x_1 and x_2 in a similar way as a pruning does. Given a function $\mu(x)$ that agrees with the min-weight $\omega(x)$ on positive instances x, this leads to the following predicate T satisfying the requirements (16) and (17) on instances (z^*, x) where z^* has a unique solution y^*:

$$T(z^*,x) \Leftrightarrow \begin{cases} \omega(z^* \sqcup x, \tau(z^*,x,1,y^*)) \neq \mu(z^* \sqcup x) & \text{for } z^* \leq_{lex} x \\ \omega(x \sqcup z^*, \tau(x,z^*,2,y^*)) \neq \mu(x \sqcup z^*) & \text{for } x \leq_{lex} z^*. \end{cases} \tag{20}$$

As in the setting of part 1, we obtain an efficient algorithm with polynomial advice for deciding $L(\Pi)$ on instances with at most one solution. To handle all inputs, we no longer have access to a pruning as we did in the case of part 1 of the theorem. However, whereas access to the functions ω and μ does not immediately yield an efficient pruning, it does yield an efficient disambiguation in case the search for a solution of a given weight can be efficiently reduced to Π under a reduction that preserves the number of solutions. This is the

case for each of Satisfiability, Reachability, and Circuit Certification on shallow semi-unbounded circuits, both for polynomial-time efficiency and for logspace efficiency.

This completes the argument for parts 1 and 2 of Theorem 7 (as well as their counterparts for Circuit Certification) in the case where the pruning f and the functions ω and μ are deterministic. For the more general case where they can be randomized and have probability of success at least $\frac{2}{3} + \frac{1}{\text{poly}(n)}$, some additional properties of Π are needed and the predicate T has to be generalized in the appropriate way. The following lemma captures the general case. We view a randomized mapping as a determinstic one that gets a random bit string $\rho \in \{0,1\}^r$ as an additional input, and often write ρ as a subscript to the name of the procedure.

We state the lemma for logspace efficiency for concreteness, but the proof only requires mild properties of the underlying notion of efficiency. In particular, it also applies to polynomial-time efficiency.

▶ **Lemma 23.** *Let $\Pi : X \mapsto 2^Y$ be a computational problem with an underlying predicate π that is computable in logspace and has the following additional properties:*

- *Π has a disjoint union operator given by $\sqcup$ and τ in Definition 21 where $\sqcup$ and τ are computable in logspace.*
- *Π has a randomized disambiguation g with probability of success at least $1/\text{poly}(n)$ that is computable and recoverable in logspace.*
- *There exists a logspace mapping reduction h from the following decision problem to Π: On input an instance $x \in X$ and an index $i \in \mathbb{N}$, decide whether there exists $y \in \Pi(x)$ such that the ith bit of y is 1. Furthermore, the instances $h(x,i)$ have at most one solution if the instance x does, and there exists a constant c such that the solutions to instances of Π of size n are strings of length n^c.*

For any $p = \frac{2}{3} + \frac{1}{\text{poly}(n)}$ either of the following hypotheses imply that $L(\Pi 0$ can be decided in logspace with polynomial advice, where ρ is chosen uniformly at random from $\{0,1\}^r$ for some $r = \text{poly}(n)$:

1. *There exists a randomized mapping $f : X \mapsto X$ computable in logspace such that for every input $x \in X$:*

$$\Pr_\rho[\ f_\rho \text{ satisfies the pruning requirement on input } x\ ] \geq p. \tag{21}$$

2. *There exist randomized mappings $\omega : X \times Y \times \mapsto \mathbb{N}$ and $\mu : X \mapsto \mathbb{N}$ that are computable in space $O(\log n)$ such that for every $x \in L(\Pi)$*

$$\Pr_\rho[\ \omega_\rho(x,\cdot) \text{ is min-isolating for } x \text{ and } \mu_\rho(x) = \omega_\rho(x)\ ] \geq p. \tag{22}$$

Proof. Let us first focus on the instances of Π that have at most one solution. Consider the predicate T defined as follows on input (z^*, x) where$\Pi(z^*) = \{y^*\}$ and q denotes a fraction to be set:

$$T(z^*,x) \Leftrightarrow \begin{cases} \Pr_\rho[\text{ right-hand side of (19) holds }] > q & \text{for part 1} \\ \Pr_\rho[\text{ right-hand side of (20) holds }] > q & \text{for part 2,} \end{cases} \tag{23}$$

where $\rho \in \{0,1\}^r$ is chosen uniformly at random for some $r = \text{poly}(n)$, and is used as the randomness for all randomized mappings involved.

▶ **Claim 24.** *Both (16) and (17) hold for $q = 1/3$ as long as $p > 2/3$, where D represents the set of all instances of Π with at most one solution.*

Proof. We argue by contradiction that T satisfies condition (16). Consider part 1 first, and suppose that neither $T(x, z^*)$ nor $T(z^*, x)$ hold for some $x, z^* \in D \cap L(\Pi)$. Then with probability at most $2q$ the translation of the unique solution for at least one of x or z^* is not a solution for $f(x^*)$ where $x^* \doteq \min(x, z^*) \sqcup \max(x, z^*)$ and max and min refer to the lexicographic order $\leq_{lex}$. By complementing, with probability at least $1 - 2q$ it is the case that both are solutions for $f(x^*)$, which therefore has at least two distinct solutions. Thus, f fails the pruning condition on input x^* with probability at least $1 - 2q$, which contradicts the hypothesis that f has success probability p as long as $q < p/2$. In the case of part 2, a similar argument by contradiction leads to the conclusion that with probability at least $1 - 2q$ two distinct solutions for x^* achieve the value $\mu(x^*)$ under ω, which contradicts the hypothesis (22) as long as $q < p/2$.

We argue condition (17) by contradiction also. For part 1, consider $z^* \in D \cap L(\Pi)$ and $x \in D \setminus L(\Pi)$, and let $x^* \doteq \min(x, z^*) \sqcup \max(x, z^*)$. Note that if the right-hand side of (19) holds then f fails the pruning property on input x^*. Thus, if $T(z^*, x)$ holds, then f fails the pruning property on input x^* with probability more than q, which contradicts the hypothesis (21) as long as $q \geq 1 - p$. For part 2, a similar argument leads to a contradiction with the hypothesis (22) as long as $q \geq 1 - p$.

The conditions $q < p/2$ and $q \geq 1 - p$ imply that $p > 2/3$, which is where the bound of $2/3$ in the statement of the lemma comes from. Setting $q = 1/3$ satisfies both requirements when $p > 2/3$. This finishes the proof of Claim 24. ◀

Note that the statement of the lemma entails some leeway in that p does not just exceeds $2/3$ but does so with some margin, namely $p \geq \frac{2}{3} + \frac{1}{\text{poly}(n)}$. We now exploit this leeway to replace the randomness in the definition of T by advice. More specifically, an application of the Chernoff bound shows that a subset R of a sufficiently large polynomial number of random strings $\rho \in \{0,1\}^r$ has the following property with high probability: All of the conditions (21) (in the case of part 1) or (22) (in the case of part 2) hold for all inputs x of length n simultaneously when the uniform distribution of ρ over $\{0,1\}^r$ is replaced by the uniform distribution over R, and p is replaced by $\tilde{p}$ for some $\tilde{p} = \frac{2}{3} + \frac{1}{\text{poly}(n)}$. By fixing a good set R and giving it as advice, the predicates (23) become computable in logspace.

This shows the existence of an algorithm A that runs in logspace with polynomial advice and correctly decides $L(\Pi)$ on instances $x \in X$ with at most one solution. In order to handle *all* instances $x \in X$ we employ the randomized disambiguation g to reduce to the case of at most one solution, the predicate h to retrieve a solution in case it is unique, and the predicate π to check purported solutions.

Denoting by σ the random bit string of the randomized disambiguation g, another application of the Chernoff bound shows that for every size n there exists a set S of $\text{poly}(n)$ strings of length $\text{poly}(n)$ each such that for every instance $x \in X$ of size n there exists at least one $\sigma \in S$ such that g_σ satisfies the disambiguation requirement on input x, i.e., $x_\sigma \doteq g_\sigma(x)$ is an instance of Π that is equivalent with respect to membership to $L(\Pi)$, and has at most one solution. Thus, we can apply our algorithm A to decide $L(\Pi)$ on x_σ.

We do not know which σ works but we do know that there is at least one and that for anyone that does, the only possible solution for the instance x_σ of Π is $(L(\Pi)(h(x_\sigma, i)))_{i=1}^{n^c}$. This follows because if x_σ has a unique solution then the ith bit of that solution is 1 if and only if there exists a solution whose ith bit is 1, and by definition $h(x_\sigma, i)$ is an instance of Π whose memberhip to $L(\Pi)$ is equivalent to the latter decision. Moreover, the instances $h(x_\sigma, i)$ of Π each have at most one solution themselves, so we can use our algorithm A to

decide $L(\Pi)$ on those instances and retrieve the only candidate solution for x_σ as

$$y_\sigma \doteq (A(h(x_\sigma, i)))_{i=1}^{n^c}.$$

Finally, we try all possible $\sigma \in S$, and check whether $g'(x, x_\sigma, y_\sigma)$ is a valid solution for x, where $g' : X \times X \times Y \mapsto Y$ denotes the logspace recovery algorithm underlying the recoverable disambiguation g. More formally, we evaluate the predicate

$$\bigvee_{\sigma \in S} \pi(x, g'(x, x_\sigma, y_\sigma)). \tag{24}$$

If $x \in L(\Pi)$ then we know that for at least one $\sigma \in S$, y_σ is the unique solution to x_σ, and $g'(x, x_\sigma, y_\sigma)$ is a valid solution to x, so (24) evaluates to true. If $x \notin L(\Pi)$, then there is no string y for which $\pi(x, y)$ holds, so (24) evaluates to false no matter what. Thus, (24) correctly decides $L(\Pi)$ on all instances $x \in X$. As all the algorithms involved run in logspace with access to their random bit strings, which are given as advice, it follows that the predicate (24) can be evaluated in logspace with polynomial advice. This concludes the proof of Lemma 23. ◀

Theorem 7 follows from an instantiation of Lemma 23 with Reachability on layered digraphs as the computational problem Π.

Proof of Theorem 7. Since Reachability on layered digraphs is hard for NL under logspace mapping reductions (see Proposition 10), it suffices to verify that Reachability on layered digraphs has all the properties required of the computational problem Π in Lemma 23. The properties regarding the predicate π and the disjoint union operator follow from Proposition 22. The existence of the required randomized disambiguation g follows from the Isolation Lemma (as explained in the introduction). Finally, here is how we can compute the required retrieving predicate $h(x, i)$ for $x \doteq (G, s, t)$. The index i corresponds to a bit position, say the jth one, of the label of an edge in some layer, say the ℓth one, of G. The instance $h(x, i)$ is obtained by removing from G all edges in layer ℓ whose jth bit is not 1. This operation can be performed in logspace. ◀

A similar argument for Circuit Certification on shallow layered alternating semi-unbounded circuits yields the following equivalent to Theorem 7.

▶ **Theorem 25.** *Either of the following hypotheses imply that* LogCFL $\subseteq$ L/poly*:*

1. Circuit Certification *on shallow layered alternating semi-unbounded circuits has a logspace pruning.*
2. Circuit Certification *on shallow layered alternating semi-unbounded circuits has a logspace weight function ω that is min-isolating, and there exists a logspace function μ such that $\mu(x)$ equals the min-weight $\omega(x)$ of x under ω on positive instances x.*

In fact, the conclusion holds even if the algorithms are randomized, as long as the probability of success exceeds $\frac{2}{3} + \frac{1}{\mathrm{poly}(n)}$ and the algorithms run in logspace when given two-way access to the random bits.

Acknowledgements. We thank the anonymous reviewers of the conference submission for their helpful suggestions.

References

1 A. Aggarwal, R. J. Anderson, and M.-Y. Kao. Parallel depth-first search in general directed graphs. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing*, pages 297–308, 1989. doi:10.1145/73007.73035.

2 M. Agrawal, R. Gurjar, A. Korwar, and N. Saxena. Hitting-sets for ROABP and sum of set-multilinear circuits. *SIAM Journal on Computing*, 44(3):669–697, 2015. doi:10.1137/140975103.

3 E. Allender and U. Hertrampf. Depth reduction for circuits of unbounded fan-in. *Information and Computation*, 112(2):217–238, 1994.

4 E. Allender, K. Reinhardt, and S. 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.

5 R. Arora, A. Gupta, R. Gurjar, and R. Tewari. Derandomizing Isolation Lemma for $K_{3,3}$-free and K_5-free Bipartite Graphs. In *Proceedings of the 33rd Symposium on Theoretical Aspects of Computer Science*, pages 10:1–10:15, 2016. doi:10.4230/LIPIcs.STACS.2016.10.

6 V. Arvind and P. Mukhopadhyay. Derandomizing the isolation lemma and lower bounds for circuit size. In *Proceedings of the 12th Intl. Workshop on Randomization and Computation*, pages 276–289, 2008. doi:10.1007/978-3-540-85363-3_23.

7 V. Arvind, P. Mukhopadhyay, and S. Srinivasan. New results on noncommutative and commutative polynomial identity testing. *Computational Complexity*, 19(4):521–558, 2010. doi:10.1007/s00037-010-0299-8.

8 G. Barnes, J. F. Buss, W. L. Ruzzo, and B. Schieber. A sublinear space, polynomial time algorithm for directed s-t connectivity. *SIAM Journal on Computing*, 27(5):1273–1282, 1998.

9 R. Beigel, N. Reingold, and D. Spielman. The perceptron strikes back. In *Proceedings of the Sixth Annual Structure in Complexity Theory Conference*, pages 286–291, 1991. doi:10.1109/SCT.1991.160270.

10 S. Ben-David, B. Chor, O. Goldreich, and M. Luby. On the theory of average case complexity. *Journal of Computer and System Sciences*, 44(2):193–219, 1992. doi:10.1016/0022-0000(92)90019-F.

11 A. Björklund. Determinant sums for undirected hamiltonicity. *SIAM Journal on Computing*, 43(1):280–299, 2014. doi:10.1137/110839229.

12 A. Björklund and T. Husfeldt. Shortest two disjoint paths in polynomial time. In *Proceedings of the 41st International Colloquium on Automata, Languages, and Programming*, pages 211–222, 2014. doi:10.1007/978-3-662-43948-7_18.

13 C. Bourke, R. Tewari, and N. V. Vinodchandran. Directed planar reachability is in unambiguous log-space. *ACM Transactions on Computation Theory*, 1(1), 2009. doi:10.1145/1490270.1490274.

14 T. Brunsch, K. Cornelissen, B. Manthey, and H. Röglin. Smoothed analysis of belief propagation for minimum-cost flow and matching. In *Proceedings of the 7th International Workshop on Algorithms and Computation*, pages 182–193, 2013. doi:10.1007/978-3-642-36065-7_18.

15 J.-Y. Cai, V. T. Chakaravarthy, and D. van Melkebeek. Time-space tradeoff in derandomizing probabilistic logspace. *Theory Comput. Syst.*, 39(1):189–208, 2006. doi:10.1007/s00224-005-1264-9.

16 C. Calabro, R. Impagliazzo, V. Kabanets, and R. Paturi. The complexity of unique k-SAT: an isolation lemma for k-CNFs. In *Proceedings of the 18th Annual IEEE Conference on Computational Complexity*, pages 135–141, 2003. doi:10.1109/CCC.2003.1214416.

17 J. L. Carter and M. N. Wegman. Universal classes of hash functions. *Journal of Computer and System Sciences*, 18(2):143–154, 1979. doi:10.1016/0022-0000(79)90044-8.

18 S. Chari, P. Rohatgi, and A. Srinivasan. Randomness-optimal unique element isolation with applications to perfect matching and related problems. *SIAM Journal on Computing*, 24(5):1036–1050, 1995. doi:10.1137/S0097539793250330.

19 M. Cygan, S. Kratsch, and J. Nederlof. Fast hamiltonicity checking via bases of perfect matchings. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing*, pages 301–310, 2013. doi:10.1145/2488608.2488646.

20 M. Cygan, J. Nederlof, M. Pilipczuk, M. Pilipczuk, J. M. M. van Rooij, and J. O. Wojtaszczyk. Solving connectivity problems parameterized by treewidth in single exponential time. In *Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science*, pages 150–159, 2011. doi:10.1109/FOCS.2011.23.

21 S. I. Daitch and D. A. Spielman. Faster approximate lossy generalized flow via interior point algorithms. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, pages 451–460, 2008. doi:10.1145/1374376.1374441.

22 S. Datta, W. Hesse, and R. Kulkarni. Dynamic complexity of directed reachability and other problems. In *Proceedings of the 41st International Colloquium on Automata, Languages, and Programming*, pages 356–367, 2014. doi:10.1007/978-3-662-43948-7_30.

23 S. Datta, R. Kulkarni, A. Mukherjee, T. Schwentick, and T. Zeume. Reachability is in DynFO. In *Proceedings of the 42nd International Colloquium on Automata, Languages, and Programming*, pages 159–170, 2015. doi:10.1007/978-3-662-47666-6_13.

24 S. Datta, R. Kulkarni, and S. Roy. Deterministically isolating a perfect matching in bipartite planar graphs. *Theory Comput. Syst.*, 47(3):737–757, 2010. doi:10.1007/s00224-009-9204-8.

25 S. Datta, R. Kulkarni, R. Tewari, and N. V. Vinodchandran. Space complexity of perfect matching in bounded genus bipartite graphs. *Journal of Computer and System Sciences*, 78(3):765–779, 2012.

26 H. Dell, V. Kabanets, D. van Melkebeek, and O. Watanabe. Is Valiant-Vazirani's isolation probability improvable? *Computational Complexity*, 22(2):345–383, 2013. doi:10.1007/s00037-013-0059-7.

27 J. Erickson and P. Worah. Computing the shortest essential cycle. *Discrete and Computational Geometry*, 44(4):912–930, 2010. doi:10.1007/s00454-010-9241-8.

28 S. A. Fenner, R. Gurjar, and T. Thierauf. Bipartite perfect matching is in quasi-NC. In *Proceedings of the 48th Annual ACM Symposium on Theory of Computing*, pages 754–763, 2016. doi:10.1145/2897518.2897564.

29 F. V. Fomin and P. Kaski. Exact exponential algorithms. *Communications of the ACM*, 56(3):80–88, March 2013. doi:10.1145/2428556.2428575.

30 A. Gál and A. Wigderson. Boolean complexity classes vs. their arithmetic analogs. *Random Struct. Algorithms*, 9(1-2):99–111, 1996. doi:10.1002/(SICI)1098-2418(199608/09)9:1/2<99::AID-RSA7>3.0.CO;2-6.

31 D. Gamarnik, D. Shah, and Y. Wei. Belief propagation for min-cost network flow: Convergence and correctness. *Operations Research*, 60(2):410–428, 2012. doi:10.1287/opre.1110.1025.

32 I. Haviv and O. Regev. On the lattice isomorphism problem. In *Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 391–404, 2014. URL: http://dl.acm.org/citation.cfm?id=2634074.2634103.

33 H. Hirai and H. Namba. Shortest $(A + B)$-path packing via hafnian. *Computing Research Repository*, abs/1603.08073, 2016. URL: http://arxiv.org/abs/1603.08073.

34 V. A. T. Kallampally and R. Tewari. Trading Determinism for Time in Space Bounded Computations. In *Proceedings of the 41st International Symposium on Mathematical Foundations of Computer Science*, pages 10:1–10:13, 2016. doi:10.4230/LIPIcs.MFCS.2016.10.

35 R. Kannan, H. Venkateswaran, V. Vinay, and A.C. Yao. A circuit-based proof of Toda's theorem. *Information and Computing*, 104(2):271–276, 1993.

36 Y. Kanoria, M. Bayati, C. Borgs, J. Chayes, and A. Montanari. Fast convergence of natural bargaining dynamics in exchange networks. In *Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1518–1537, 2011. URL: http://dl.acm.org/citation.cfm?id=2133036.2133154.

37 R. M. Karp, E. Upfal, and A. Wigderson. Constructing a perfect matching is in random nc. *Combinatorica*, 6(1):35–48, 1986. doi:10.1007/BF02579407.

38 S. Kiefer, A. S. Murawski, J. Ouaknine, B. Wachter, and J. Worrell. On the complexity of equivalence and minimisation for Q-weighted automata. *Logical Methods in Computer Science*, 9(1), 2013. doi:10.2168/LMCS-9(1:8)2013.

39 A. Klivans and D. A. Spielman. Randomness efficient identity testing of multivariate polynomials. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing*, pages 216–223, 2001. doi:10.1145/380752.380801.

40 A. Klivans and D. 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.

41 K. Ko. On self-reducibility and weak P-selectivity. *Journal of Computer and System Sciences*, 26(2):209–211, 1983.

42 V. Krishan and N. Limaye. Isolation lemma for directed reachability and NL vs. L. *Electronic Colloquium on Computational Complexity*, 23:155, 2016. URL: http://eccc.hpi-web.de/report/2016/155.

43 J. Kynčl and T. Vyskočil. Logspace reduction of directed reachability for bounded genus graphs to the planar case. *ACM Transactions on Computation Theory*, 1(3):8:1–8:11, March 2010. doi:10.1145/1714450.1714451.

44 A. Lingas and M. Karpinski. Subtree isomorphism is NC reducible to bipartite perfect matching. *Information Processing Letters*, 30(1):27–32, 1989. doi:10.1016/0020-0190(89)90170-1.

45 A. Lingas and M. Persson. A fast parallel algorithm for minimum-cost small integral flows. *Algorithmica*, 72(2):607–619, 2015. doi:10.1007/s00453-013-9865-1.

46 R. Majumdar and J. L. Wong. Watermarking of SAT using combinatorial isolation lemmas. In *Proceedings of the 38th Annual Design Automation Conference*, pages 480–485, 2001. doi:10.1145/378239.378566.

47 K. Mulmuley, U. V. Vazirani, and V. V. Vazirani. Matching is as easy as matrix inversion. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing*, pages 345–354, 1987. doi:10.1145/28395.383347.

48 N. Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992. doi:10.1007/BF01305237.

49 N. Nisan. RL *subseteq* SC. In *Proceedings of the 24th Annual ACM Symposium on Theory of Computing, May 4-6, 1992, Victoria, British Columbia, Canada*, pages 619–623, 1992. doi:10.1145/129712.129772.

50 J. B. Orlin and C. Stein. Parallel algorithms for the assignment and minimum-cost flow problems. *Operations research letters*, 14(4):181–186, 1993.

51 K. Reinhardt and E. Allender. Making nondeterminism unambiguous. *SIAM Journal on Computing*, 29(4):1118–1131, 2000. doi:10.1137/S0097539798339041.

52 M. E. Saks and S. Zhou. $\mathrm{BP}_H\mathrm{SPACE}(S) \subseteq \mathrm{DSPACE}(S^{3/2})$. *Journal of Computer and System Sciences*, 58(2):376–403, 1999. doi:10.1006/jcss.1998.1616.

53 W. J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *Journal of Computer and System Sciences*, 4(2):177–192, 1970. doi:10.1016/S0022-0000(70)80006-X.

54 Y. Strozecki. On enumerating monomials and other combinatorial structures by polynomial interpolation. *Theory of Computing Systems*, 53(4):532–568, 2013. doi:10.1007/s00224-012-9442-z.

55 I. H. Sudborough. On the tape complexity of deterministic context-free languages. *Journal of the ACM*, 25(3):405–414, July 1978. doi:10.1145/322077.322083.

56 J. Tarui. Probabilistic polynomials, AC0 functions and the polynomial-time hierarchy. *Theoretical Computer Science*, 113(1):167–183, 1993.

57 T. Thierauf and F. Wagner. Reachability in $K_{3,3}$-free and K_5-free graphs is in unambiguous logspace. *Chicago Journal of Theoretical Computer Science*, 2015, 2015. URL: http://cjtcs.cs.uchicago.edu/articles/2015/2/contents.html.

58 S. Toda. PP is as hard as the polynomial-time hierarchy. *SIAM Journal on Computing*, 20(5):865–877, 1991. doi:10.1137/0220053.

59 P. Traxler. The time complexity of constraint satisfaction. In *Proceedings of the 3rd International Workshop on Parameterized and Exact Computation*, pages 190–201, 2008. doi:10.1007/978-3-540-79723-4_18.

60 L. G. Valiant and V. V. Vazirani. NP is as easy as detecting unique solutions. *Theoretical Computer Science*, 47(3):85–93, 1986. doi:10.1016/0304-3975(86)90135-0.

61 H. Venkateswaran. Properties that characterize LOGCFL. *Journal of Computer and System Sciences*, 43(2):380–404, October 1991. doi:10.1016/0022-0000(91)90020-6.

62 H. Vollmer. *Introduction to Circuit Complexity: A Uniform Approach*. Springer-Verlag New York, 1999.

63 O. Watanabe and S. Toda. Structural analysis of the complexity of inverse functions. *Mathematical Systems Theory*, 26(2):203–214, 1993.

The Computational Complexity of Integer Programming with Alternations*

Danny Nguyen[1] and Igor Pak[2]

1 **Department of Mathematics, UCLA, Los Angeles, CA, USA**
ldnguyen@math.ucla.edu
2 **Department of Mathematics, UCLA, Los Angeles, CA, USA**
pak@math.ucla.edu

Abstract

We prove that integer programming with three alternating quantifiers is NP-complete, even for a fixed number of variables. This complements earlier results by Lenstra and Kannan, which together say that integer programming with at most two alternating quantifiers can be done in polynomial time for a fixed number of variables. As a byproduct of the proof, we show that for two polytopes $P, Q \subset \mathbb{R}^4$, counting the projection of integer points in $Q \setminus P$ is #P-complete. This contrasts the 2003 result by Barvinok and Woods, which allows counting in polynomial time the projection of integer points in P and Q separately.

1998 ACM Subject Classification F.1.3 Complexity Measures and Classes, G.1.6 Optimization

Keywords and phrases integer programming, alternations, projection of integer points

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.6

1 Introduction

1.1 Background

In a pioneer paper [19], Lenstra showed that Integer Programming in a bounded dimension can be solved in polynomial time. The next breakthrough was obtained by Kannan in 1990 and until recently remained the most general result in this direction (see [11]).

▶ **Theorem 1** (Parametric Integer Programming [16]). *Fix d_1 and d_2. Given a polyhedron $P \subseteq \mathbb{R}^{d_1}$, a matrix $A \in \mathbb{Z}^{m \times (d_1+d_2)}$ and a vector $\overline{b} \in \mathbb{Z}^m$, the following sentence can be decided in polynomial time:*

$$\forall \mathbf{x} \in P \cap \mathbb{Z}^{d_1} \quad \exists \mathbf{y} \in \mathbb{Z}^{d_2} \quad : \quad A(\mathbf{x}, \mathbf{y}) \le \overline{b}. \tag{1.1}$$

Here P is given by a system $C\mathbf{x} \le \overline{\gamma}$, with $C \in \mathbb{Z}^{n \times d_1}$ and $\overline{\gamma} \in \mathbb{Z}^n$. The numbers m, n are part of the input.

In [17], Kannan asked if Theorem 1 can be extended to three alternating quantifiers. We give an answer in the negative direction to this question:

▶ **Theorem 2.** *Fix $d_1 \ge 1, d_2 \ge 2$ and $d_3 \ge 3$. Given two polyhedra $P \subseteq \mathbb{R}^{d_1}$, $Q \subseteq \mathbb{R}^{d_2}$, a matrix $A \in \mathbb{Z}^{m \times (d_1+d_2+d_3)}$ and a vector $\overline{b} \in \mathbb{Z}^m$, then deciding the sentence*

$$\exists \mathbf{x} \in P \cap \mathbb{Z}^{d_1} \quad \forall \mathbf{y} \in Q \cap \mathbb{Z}^{d_2} \quad \exists \mathbf{z} \in \mathbb{Z}^{d_3} \quad : \quad A(\mathbf{x}, \mathbf{y}, \mathbf{z}) \le \overline{b} \tag{1.2}$$

is an NP-complete problem. Here P and Q are given by two systems $C\mathbf{x} \le \overline{\gamma}$ and $D\mathbf{y} \le \overline{\delta}$, with $C \in \mathbb{Z}^{n \times d_1}$, $\overline{\gamma} \in \mathbb{Z}^n$, $D \in \mathbb{Z}^{q \times d_2}$, and $\overline{\delta} \in \mathbb{Z}^q$.

* The second author was partially supported by the NSF.

Let us emphasize that in both Theorem 1 and 2, there is no bound on the number of inequalities involved. In other words, the parameters m, n and q are *not* fixed. Theorem 2 is especially surprising for the following reasons. First, in [22], we gave strong evidence that (1.2) is decidable in polynomial time if m, n and q are fixed. Second, by an easy application of the Doignon–Bell–Scarf theorem, (1.1) is polynomial time reducible to the case with m and n fixed. Unfortunately, this simple reduction breaks down when there are more than two quantifiers (see Section 7.1) as in (1.2). Still, in [22], we speculated that a more involved reduction argument might still apply to (1.2). Theorem 2 refutes the possibility of any reduction from (1.2) to an easier form with m, n and q bounded for which decision could be in polynomial time, unless $\mathsf{P} = \mathsf{NP}$. In fact, Theorem 2 holds even when P is an interval and Q is an axis-parallel rectangles (see Theorem 9 and §7.8). Thus, the problem (1.2) is already hard when n, q are fixed and only m is unbounded.

In [25], Schöning proved that it is NP-complete to decide whether

$$\exists x \in \mathbb{Z} \quad \forall y \in \mathbb{Z} \quad : \quad \Psi(x, y). \tag{1.3}$$

Compared to (1.2), this has only two quantifiers. However, here the expression $\Psi(x, y)$ is allowed to contain both conjunctions and disjunctions of many inequalities. So Theorem 2 tells us that disjunctions can be discarded at the cost of adding one extra alternation. In the next subsection, we generalize this observation.

1.2 Presburger sentences

In [14], Grädel considered the theory of *Presburger Arithmetic*, and proved many completeness results in this theory when the number of variables and quantifiers are bounded. Those results were later strengthened by Schöning in [25]. They can be summed up as follows:

▶ **Theorem 3** ([25]). *Fix $k \geq 1$. Let $\Psi(\mathbf{x}, \mathbf{y})$ be a Boolean combination of linear inequalities with integer coefficients in the variables $\mathbf{x} = (x_1, \ldots, x_k) \in \mathbb{Z}^k$ and $\mathbf{y} = (y_1, \ldots, y_3) \in \mathbb{Z}^3$. Then deciding the sentence*

$$Q_1\, x_1 \in \mathbb{Z} \quad \ldots \quad Q_k\, x_k \in \mathbb{Z} \quad Q_{k+1}\, \mathbf{y} \in \mathbb{Z}^3 \quad : \quad \Psi(\mathbf{x}, \mathbf{y})$$

is $\mathbf{\Sigma}_k^{\mathsf{P}}$-complete if $Q_1 = \exists$, and $\mathbf{\Pi}_k^{\mathsf{P}}$-complete if $Q_1 = \forall$. Here $Q_1, \ldots, Q_{k+1} \in \{\forall, \exists\}$ are $m + 1$ alternating quantifiers.

This result characterizes the complexity of so called *Presburger sentences* with $k + 1$ quantifiers in a fixed number of variables. The main difference between Presburger Arithmetic versus integer programming is that the expression Ψ allows both conjunction and disjunction of many inequalities. This flexibility allows effective reductions of classical decision problems such as QSAT. For some time, it remains a question whether such reductions can be carried with only conjunctions, and at the same time keeping the number of variables fixed. We prove the following result, which generalizes Theorem 2:

▶ **Theorem 4.** *Integer programming in a fixed number of variables with $k + 2$ alternating quantifiers is $\mathbf{\Sigma}_k^{\mathsf{P}}/\mathbf{\Pi}_k^{\mathsf{P}}$-complete, depending on whether $Q_1 = \exists/\forall$. Here the problem is allowed to contain only a system of inequalities.*

We refer to Theorem 13 for the precise statement. Thus, we see that integer programming requires only one more quantifier alternation to achieve the same complexity as Presburger Arithmetic. Again, we emphasize that while the number of variables and quantifiers are fixed in Theorem 4, the linear system is still allowed many inequalities.

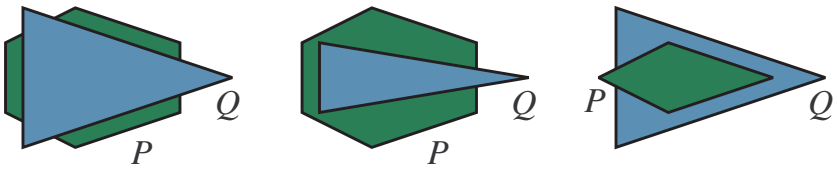


Figure 1 Three examples of convex polygons $P, Q \subset \mathbb{R}^2$.

1.3 Counting points in projections of non-convex polyhedra

For polytopes in arbitrary dimension, counting the number of integer points is classically #P-complete, even for 0/1 polytopes. In a fixed dimension d, Barvinok famously showed this can be done in polynomial time:

▶ **Theorem 5** ([2]). *Fix d. Given a polytope $P \subset \mathbb{R}^d$, the number of integer points in $P \cap \mathbb{Z}^d$ can be computed in polynomial time. Here P is described by a system $A\mathbf{x} \le \bar{b}$, with $A \in \mathbb{Z}^{m\times d}$, $\bar{b} \in \mathbb{Z}^m$.*

For a set $S \subset \mathbb{R}^d$, denote by $\mathrm{E}(S) := S \cap \mathbb{Z}^d$. The previous results say that $|\mathrm{E}(P)|$ is computable in polynomial time. Given two polytopes $P \subset Q \subset \mathbb{R}^d$, we clearly have $|\mathrm{E}(Q\setminus P)| = |\mathrm{E}(Q)| - |\mathrm{E}(P)|$. So the number of integer points in a complement can also be computed effectively.

Theorem 5 was later generalized by Barvinok and Woods to count the number of integer points in projections of polytopes:

▶ **Theorem 6** ([5]). *Fix d_1 and d_2. Given a polytope $P \subset \mathbb{R}^{d_1}$, and a linear transformation $T : \mathbb{Z}^{d_1} \to \mathbb{Z}^{d_2}$, the number of integer points in $T(P \cap \mathbb{Z}^{d_1})$ can be computed in polynomial time. Here P is described by a system $A\mathbf{x} \le \bar{b}$ and T is described by a matrix M, where $A \in \mathbb{Z}^{m\times d_1}$, $\bar{b} \in \mathbb{Z}^m$ and $M \in \mathbb{Z}^{d_2\times d_1}$.*

For a set $S \subset \mathbb{R}^d$, denote by $\mathrm{E}_1(S)$ the projection of $S \cap \mathbb{Z}^d$ on the first coordinate, i.e.,

$$\mathrm{E}_1(S) := \{x \in \mathbb{Z} \quad : \quad \exists \mathbf{z} \in \mathbb{Z}^{d-1} \quad (x, \mathbf{z}) \in S\}.$$

By Theorem 6, $|\mathrm{E}_1(P)|$ can be computed in polynomial time for every polytope $P \subset \mathbb{R}^d$.

We prove the following result:

▶ **Theorem 7.** *Given two polytopes $P \subset Q \subset \mathbb{R}^4$, computing $|\mathrm{E}_1(Q\setminus P)|$ is #P-complete.*

In other words, it is #P-complete to compute the size of the set

$$\mathrm{E}_1(Q\setminus P) = \{x \in \mathbb{Z} \,:\, \exists \mathbf{z} \in \mathbb{Z}^3 \quad (x, \mathbf{z}) \in Q\setminus P\}. \tag{1.4}$$

Note that the corresponding decision problem $|\mathrm{E}_1(Q\setminus P)| \ge 1$ is equivalent to $|\mathrm{E}(Q\setminus P)| \ge 1$, and thus can be decided in polynomial time by applying Theorem 5.

The contrast between Theorem 6 and our negative result can be explained as follows. The proof Theorem 6 depends on the polytopal structure of P and exploited convexity in a crucial way. By taking the complement $Q\setminus P$, we no longer have a convex set. In other words, we show that projection of the complement $Q\setminus P$ is complicated enough to allow encoding of hard counting problems, even in $\mathbb{R}^4$ (see also §7.5).

▶ Remark 1. To understand the theorem, consider three examples of polygons $P, Q \subset \mathbb{R}^2$ as in Figure 1. Note that the sets of integer points of the vertical projections of P, Q and $P \cup Q$ are the same in all three cases, but the sets number of integer points of the vertical projections of $Q \setminus P$ are quite different.

As an easy consequence of Theorem 7 we obtain:

▶ **Corollary 8.** *Given r simplices $T_1, \ldots, T_r \subset \mathbb{R}^4$, computing $|\mathrm{E}_1(T_1 \cup \cdots \cup T_r)|$ is #P-complete.*

1.4 Outline of the paper

We begin with notations (Section 2) and a geometric construction of certain polytopes based on Fibonacci numbers (Section 3). In Section 4 we use this construction to prove Theorem 2 via a reduction of the GOOD SIMULTANEOUS APPROXIMATION (GSA) Problem in Number Theory, which is known to be NP-complete. The proof of Theorem 4 is via a reduction of QSAT (Section 5). The proof of Theorem 7 follows a similar route via reduction of #GSA (Section 6). Finally, we conclude with final remarks and open problems (Section 7).

2 Notations

- We a use $\mathbb{N} = \{0, 1, 2, \ldots\}$ and $\mathbb{Z}_+ = \{1, 2, \ldots\}$.
- All constant vectors are denoted $\overline{a}, \overline{b}, \overline{x}, \overline{y}, \overline{t}$ etc.
- Matrices are denoted A, B, C, etc.
- Variables are denoted x, y, z, etc.; vectors of variables are denoted $\mathbf{x}, \mathbf{y}, \mathbf{z}$, etc.
- We write $\mathbf{x} \le \mathbf{y}$ if $x_j \le y_j$ for all i.
- A *polyhedron* is an intersection of finitely many closed half-spaces in $\mathbb{R}^n$.
- A *polytope* is a bounded polyhedron.
- Polyhedra and polytopes are denoted by P, Q, R, etc.

3 Geometric constructions and properties

3.1 Fibonacci points

We consider the first $2d$ *Fibonacci numbers*:

$$F_0 = 0, F_1 = 1, F_2 = 1, \ldots, F_{2d-1}.$$

From these, we construct d integer points:

$$\phi_1 = (F_1, F_0),\ \phi_2 = (F_3, F_2),\ \ldots,\ \phi_d = (F_{2d-1}, F_{2d-2}). \tag{3.1}$$

Let

$$\Phi = \{\phi_1, \ldots, \phi_d\} \subset \mathbb{Z}^2 \quad \text{and} \quad J = [1, F_{2d-1}] \times [0, F_{2d-2}] \cap \mathbb{Z}^2. \tag{3.2}$$

We have $\Phi \subset J$. Denote by $\mathcal{C}$ the curve consisting of $d-1$ segments connecting ϕ_i to ϕ_{i+1} for $i = 1, \ldots, i-1$.

We also define the following two polygons. Their properties will be mentioned later.

$$R_1 = \left\{ \mathbf{y} = (y_1, y_2) \in \mathbb{R}^2 \ : \ \begin{bmatrix} y_1 & \ge & 1 \\ y_2 & \le & F_{2d-2} \\ y_2 F_{2d-1} - y_1 F_{2d-2} & \ge & 1 \end{bmatrix} \right\}, \tag{3.3}$$

and

$$R_2 = \left\{ \mathbf{y} \in \mathbb{R}^2 \ : \ \begin{bmatrix} y_1 \leq F_{2d-1} \\ y_2 \geq 0 \end{bmatrix} \text{ and } \ y_2 F_{2i} - y_1 F_{2i-1} \leq -2 \ \text{ for } \ i = 1, \ldots, d \right\}. \tag{3.4}$$

The following properties are straightforward from the above definitions:

(F1) The points $\phi_1, \ldots, \phi_d$ are in convex position. The curve $\mathcal{C}$ connecting them is convex (upwards). See Figure 2.

(F2) Each segment $(\phi_i\, \phi_{i+1})$ and each triangle $\Delta_i = (0\, \phi_i\, \phi_{i+1})$ has no interior integer points. This can be deduced from the facts that two consecutive Fibonacci numbers are coprime, and also

$$F_i F_{i+3} - F_{i+1} F_{i+2} = (-1)^{i-1} \quad \text{for all } i \geq 0.$$

(F3) The set of integer points in $J \setminus \Phi$ can be partitioned into 2 parts: those lying strictly above the convex curve $\mathcal{C}$, and those lying strictly below it.

(F4) The part of $J \setminus \Phi$ lying above $\mathcal{C}$ is exactly $R_1 \cap \mathbb{Z}^2$. This can be seen as follows. The line ℓ connecting 0 and ϕ_d is defined by:

$$y_2 F_{2d-1} - y_1 F_{2d-2} = 0.$$

So every integer point $\mathbf{y} = (y_1, y_2)$ lying above ℓ satisfies:

$$y_2 F_{2d-1} - y_1 F_{2d-2} \geq 1.$$

By property (F2), there are no integer points $\mathbf{y}$ between $\mathcal{C}$ and ℓ. The other two edges of R_1 come from J. See Figure 2.

(F5) The part of $J \setminus \Phi$ lying below $\mathcal{C}$ is exactly $R_2 \cap \mathbb{Z}^2$. This can be seen as follows. The line connecting ϕ_i and ϕ_{i+1} is defined by

$$y_2 F_{2i} - y_1 F_{2i-1} = -1.$$

So all integer points below that line satisfies:

$$y_2 F_{2i} - y_1 F_{2i-1} \leq -2.$$

This gives $d-1$ faces for R_2, one for each $1 \leq i \leq d-1$. The other two faces of R_2 come from J. See Figure 2.

3.2 The polytopes

Given $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_d) \in \mathbb{Q}^d$ and $\epsilon \in (0, \frac{1}{2}) \cap \mathbb{Q}$, for each $1 \leq i \leq d$, we define two polygons:

$$P_i = \{(x, w) \in \mathbb{R}^2 \quad : \quad 1 \leq x \leq N, \quad \alpha_i x - \epsilon \leq w \leq \alpha_i x + \epsilon\}, \tag{3.5}$$

and

$$Q_i = \{(x, w) \in \mathbb{R}^2 \quad : \quad 1 \leq x \leq N, \quad \alpha_i x + \epsilon - 1 < w \leq \alpha_i x + \epsilon\}. \tag{3.6}$$

Next, for each $1 \leq i \leq d$, we define two new polytopes

$$P_i' = \{(x, \phi_i, w) : (x, w) \in P_i\} \subset \mathbb{R}^4, \tag{3.7}$$

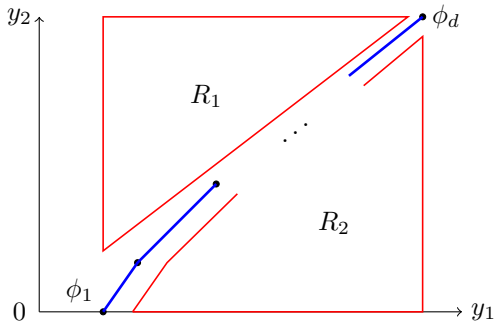


Figure 2 The points $\phi_1, \ldots, \phi_d \in \Phi$ form a convex curve $\mathcal{C}$ (blue).

and

$$Q'_i = \{(x, \phi_i, w) : (x, w) \in Q_i\} \subset \mathbb{R}^4, \tag{3.8}$$

where ϕ_i is from (3.1). Finally, we define the convex hulls:

$$P = \text{conv}(P'_1, \ldots, P'_d) \subset \mathbb{R}^4, \tag{3.9}$$

and

$$Q = \text{conv}(Q'_1, \ldots, Q'_d) \subset \mathbb{R}^4. \tag{3.10}$$

The following properties are straightforward from the above definitions:

(P1) Each P_i is a parallelogram with vertices $\big\{(1, \alpha_i \pm \epsilon),\ (N, \alpha_i N \pm \epsilon)\big\}$.

(P2) Each Q_i is a (partially open) parallelogram with vertices

$$\big\{(1, \alpha_i + \epsilon),\ (1, \alpha_i + \epsilon - 1),\ (N, \alpha_i N + \epsilon),\ (N, \alpha_i N + \epsilon - 1)\big\}.$$

(P3) Each P'_i is a *parallelogram* in $\mathbb{R}^4$ (i.e., a Minkowski sum of two intervals), with vertices $\big\{(1, \phi_i, \alpha_i \pm \epsilon),\ (N, \phi_i, \alpha_i N \pm \epsilon)\big\}$.

(P4) Each Q'_i is a (partially open) parallelogram in $\mathbb{R}^4$, with vertices

$$\big\{(1, \phi_i, \alpha_i + \epsilon),\ (1, \phi_i, \alpha_i + \epsilon - 1),\ (N, \phi_i, \alpha_i N + \epsilon),\ (N, \phi_i, \alpha_i N + \epsilon - 1)\big\}.$$

(P5) We have $P_i \subsetneq Q_i$, $P'_i \subsetneq Q'_i$ and $P \subsetneq Q$. Each P'_i forms a 2-dimensional face of P. Each Q'_i forms a 2-dimensional face of Q.

(P6) All the vertices of $P'_1, \ldots, P'_d$ are in convex position. This follows from (3.7) and (F1).

(P7) The polytope P has $4d$ vertices, which are all the vertices of $P'_1, \ldots, P'_d$. For every vertex $(x, \mathbf{y}, w)$ of P, we have $\mathbf{y} \in \Phi$, by (P3) and (P6).

(P8) For every $\phi_i \in \Phi$, we have:

$$\big\{(x, w) \in \mathbb{R}^2 : (x, \phi_i, w) \in P\big\} = P_i.$$

(P9) All the vertices $Q'_1, \ldots, Q'_d$ are also in convex position, by (3.8) and (F1).

(P10) The polytope Q has $4d$ vertices, which are all the vertices of $Q'_1, \ldots, Q'_d$. For every vertex $(x, \mathbf{y}, w)$ of Q, we have $\mathbf{y} \in \Phi$, by (P4) and (P9).

(P11) For every $\phi_i \in \Phi$, we have:

$$\big\{(x, w) \in \mathbb{R}^2 : (x, \phi_i, w) \in Q\big\} = Q_i.$$

(P12) For every point $(x, \mathbf{y}, w) \in P \cap \mathbb{Z}^4$, we have either $\mathbf{y} \in \Phi$ or $\mathbf{y} \in R_2$. This follows from (3.9), (F1) and (F5). The same holds for Q.

We will be using these properties in the latter sections.

4 Proof of Theorem 2

4.1

By a *box* in $\mathbb{Z}^d$, we mean the set of integer points of the form $[\alpha_1, \beta_1] \times \cdots \times [\alpha_d, \beta_d] \cap \mathbb{Z}^d$. We will prove the following stronger version of Theorem 2.

▶ **Theorem 9.** *Given a polytope $U \subset \mathbb{R}^6$ and two finite boxes $I \subset \mathbb{Z}$, $J \subset \mathbb{Z}^2$, deciding the sentence*

$$\exists x \in I \quad \forall \mathbf{y} \in J \quad \exists \mathbf{z} \in \mathbb{Z}^3 \quad : \quad (x, \mathbf{y}, \mathbf{z}) \in U \tag{4.1}$$

is an NP*-complete problem. Here U is described by a system $A\,(x, \mathbf{y}, \mathbf{z}) \le \overline{b}$, where $A \in \mathbb{Z}^{m \times 6}$ and $\overline{b} \in \mathbb{Z}^m$.*

Since low dimensional boxes can be easily embedded into higher dimensions, the above implies Theorem 2 for every $d_1 \ge 1, d_2 \ge 3$ and $d_3 \ge 3$. Compared to Theorem 2, all parameters in the above theorem are fixed, except for m. So from now on, the symbols n and d will be reused for other purposes. For a vector $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_d) \in \mathbb{Q}^d$ and an integer $x \in \mathbb{Z}$, we define

$$\{\{x\boldsymbol{\alpha}\}\} = \max_{1 \le i \le d} \{\{q\alpha_i\}\}, \tag{4.2}$$

where for each rational $\beta \in \mathbb{Q}$, the quantity $\{\beta\}$ is defined as:

$$\{\{\beta\}\} := \min_{n \in \mathbb{Z}} |\beta - n| = \min\big\{\beta - \lfloor\beta\rfloor, \lceil\beta\rceil - \beta\big\}.$$

GOOD SIMULTANEOUS APPROXIMATION (GSA)
Input: A rational vector $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_d) \in \mathbb{Q}^d$ and $N \in \mathbb{N}$, $\epsilon \in \mathbb{Q}$.
Decide: Is an integer $x \in [1, N]$ such that $\{\{x\boldsymbol{\alpha}\}\} \le \epsilon$?

Note that GSA is only non-trivial for $\epsilon < 1/2$. We need the following result by Lagarias:

▶ **Theorem 10** ([18])**.** GSA *is* NP*-complete.*

Let us emphasize that in GSA, the number d is part of the input. If d is fixed instead, then the problem can be decided in polynomial time (see [18] and [15, Ch. 5]). What follows is a reduction of GSA to a sentence of the form (4.1). GSA can be expressed as an integer programming problem:

$$\exists\, x, w_1, \ldots, w_d \in \mathbb{Z} \quad : \quad 1 \le x \le N, \quad -\epsilon \le \alpha_i x - w_i \le \epsilon. \tag{4.3}$$

The inequalities on w_i can be expressed as $(x, w_i) \in P_i$, where P_i was defined in (3.5). Letting $I = [1, N] \cap \mathbb{Z}$, we see that GSA is equivalent to deciding:

$$\exists x \in I \quad : \quad \bigwedge_{i=1}^{d} \Big[\exists w \in \mathbb{Z} \ : \ (x, w) \in P_i\Big]. \tag{4.4}$$

▶ **Lemma 11.** *Let $\Phi = \{\phi_1, \ldots, \phi_d\}$ be as in* (3.2) *and P be as in* (3.9)*. We have:*

$$\{\{x\boldsymbol{\alpha}\}\} \le \epsilon \quad \Longleftrightarrow \quad \forall \mathbf{y} \in \Phi \quad \exists w \in \mathbb{Z} : (x, \mathbf{y}, w) \in P. \tag{4.5}$$

Proof. Indeed, assume $\{\{x\boldsymbol{\alpha}\}\} \leq \epsilon$, i.e., x satisfies GSA. By (4.4), for every $i = 1, \ldots, d$, there exists $w_i \in \mathbb{Z}$ with $(x, w_i) \in P_i$. Now (P8) implies that $(x, \phi_i, w_i) \in P$. Since this holds for every $\phi_i \in \Phi$, the RHS in (4.5) is satisfied. For the other direction, assume the RHS in (4.5) holds. Then for every $\phi_i \in \Phi$, there exists $w_i \in \mathbb{Z}$ with $(x, \phi_i, w_i) \in P$. By (P8), we have $(x, w_i) \in P_i$. By (4.4), x satisfies GSA, i.e., $\{\{x\boldsymbol{\alpha}\}\} \leq \epsilon$. ◀

By the above lemma, GSA is equivalent to:

$$\exists x \in I \quad \forall \mathbf{y} \in \Phi \quad \exists w \in \mathbb{Z} : (x, \mathbf{y}, w) \in P. \tag{4.6}$$

Consider J from (3.2), which contains Φ. We can rewrite the above sentence as:

$$\exists x \in I \quad \forall \mathbf{y} \in J \quad \big[(\mathbf{y} \in J \backslash \Phi) \ \vee \ \exists w \in \mathbb{Z} : (x, \mathbf{y}, w) \in P\big]. \tag{4.7}$$

Recall the polygons R_1 and R_2 defined in (3.3) and (3.4). By properties (F3), (F4) and (F5), we can rewrite $\mathbf{y} \in J \backslash \Phi$ as $(\mathbf{y} \in R_1) \vee (\mathbf{y} \in R_2)$. Now, we can rewrite (4.7) as:

$$\exists x \in I \quad \forall \mathbf{y} \in J \quad \big[(\mathbf{y} \in R_1) \ \vee \ (\mathbf{y} \in R_2) \ \vee \ \exists w \in \mathbb{Z} : (x, \mathbf{y}, w) \in P\big]. \tag{4.8}$$

Next, define two polytopes R_1' and R_2' as follows:

$$R_i' := \big\{(x, \mathbf{y}, 0) \in \mathbb{R}^4 : 0 \leq x \leq N, \ \mathbf{y} \in R_i\big\} \subset \mathbb{R}^4 \quad \text{for} \quad i = 1, 2. \tag{4.9}$$

Polytopes R_1' and R_2' are defined in such a way so that for every $x \in I$ and $\mathbf{y} \in J$, we have $\mathbf{y} \in R_i$ if and only if there exists $w \in \mathbb{Z}$ such that $(x, \mathbf{y}, w) \in R_i'$.[1] Now, it is clear that (4.8) is equivalent to:

$$\exists x \in I \quad \forall \mathbf{y} \in J \quad \left[\left(\bigvee_{i=1}^{2} \exists w \in \mathbb{Z} : (x, \mathbf{y}, w) \in R_i'\right) \ \vee \ \left(\exists w \in \mathbb{Z} : (x, \mathbf{y}, w) \in P\right)\right].$$

which is equivalent to:

$$\exists x \in I \quad \forall \mathbf{y} \in J \quad \exists w \in \mathbb{Z} \quad : \quad (x, \mathbf{y}, w) \in R_1' \cup R_2' \cup P. \tag{4.10}$$

The difference between (4.10) and (4.1) is that we have 3 polytopes instead of just one.

4.2

The final step is two compress three polytopes R_1', R_2' and P into one polytope. Recall from (P7) that P has $4d$ vertices, which correspond to the vertices of all P_i for $1 \leq i \leq d$. The vertices of R_1 and R_2 can be computed in polynomial time from systems (3.3) and (3.4). From there we easily get the vertices of R_1' and R_2'. Since P, R_1' and R_2' are in the fixed dimension 4, we can write down all their facets in polynomial time using their vertices. So we can represent:

$$\begin{aligned} P &= \left\{(x, \mathbf{y}, w) \in \mathbb{R}^4 \ : \ A_1 (x, \mathbf{y}, w) \leq \bar{b}_1\right\}, \\ R_1' &= \left\{(x, \mathbf{y}, w) \in \mathbb{R}^4 \ : \ A_2 (x, \mathbf{y}, w) \leq \bar{b}_2\right\}, \\ R_2' &= \left\{(x, \mathbf{y}, w) \in \mathbb{R}^4 \ : \ A_3 (x, \mathbf{y}, w) \leq \bar{b}_3\right\}. \end{aligned} \tag{4.11}$$

The above three systems all have lengths polynomial in the input $\boldsymbol{\alpha}, N$ and ϵ. Next, we need the following lemma:

[1] Such a w must automatically be 0 by the definition of R_i'.

▶ **Lemma 12.** *Fix n and r. Given r polytopes $R_1, \ldots, R_r \subset \mathbb{R}^n$ described by r systems*

$$R_i = \{\mathbf{x} \in \mathbb{R}^n : A_i\,\mathbf{x} \le \bar{b}_i\},$$

there is a polytope $U \in \mathbb{R}^{n+\ell}$, where $\ell = \lceil \log_2 r \rceil$, such that

$$\mathbf{x} \in \bigcup_{i=1}^{r} R_i \cap \mathbb{Z}^n \quad \Longleftrightarrow \quad \exists \mathbf{t} \in \mathbb{Z}^\ell : (\mathbf{x}, \mathbf{t}) \in U \cap \mathbb{Z}^{n+\ell}. \tag{4.12}$$

Furthermore, the system $A\,(\mathbf{x}, \mathbf{t}) \le \bar{b}$ that describes U can be found in polynomial time, given A_i's and $\bar{b}_i$'s as input.

Proof. Let $\ell = \lceil \log_2 r \rceil$, we have $2^\ell \ge r$. Pick $\bar{t}_1, \ldots, \bar{t}_r \in \{0,1\}^\ell$ as r different vertices of the ℓ-dimensional unit cube. Define

$$U_j = \{(\mathbf{x}, \bar{t}_j) \in \mathbb{R}^{n+\ell} : \mathbf{x} \in R_j\} \quad \text{for} \quad j = 1, \ldots, r\,,$$

and

$$U = \operatorname{conv}(U_1, \ldots, U_r).$$

In other words, we form U_j by augmenting each R_j with ℓ coordinates of $\bar{t}_j$. Since $\bar{t}_1, \ldots, \bar{t}_r$ are in convex position, so are the new polytopes $U_1, \ldots, U_j$. So the vertices of U are all the vertices of all U_j. Note that for every $\mathbf{t} \in \operatorname{conv}(\bar{t}_1, \ldots, \bar{t}_r)$, we have $\mathbf{t} \in \mathbb{Z}^\ell$ if and only if $\mathbf{t} = \bar{t}_j$ for some j. This implies that the only integer points in U are those in U_j's. In other words:

$$(\mathbf{x}, \mathbf{t}) \in U \cap \mathbb{Z}^{n+\ell} \quad \Longleftrightarrow \quad \mathbf{x} \in R_j \cap \mathbb{Z}^n \ \text{ and } \ \mathbf{t} = \bar{t}_j \ \text{ for some } \ j = 1, \ldots, r.$$

So we have (4.12).

For each R_j, its vertices can be computed in polynomial time from the system $A_i\,\mathbf{x} \le \bar{b}_i$. From these, we easily get the vertices for each U_j. Thus, we can find all vertices of U in polynomial time. Note that U is in a fixed dimension $n+\ell$, since n and r are fixed. Therefore, we can find in polynomial time all the facets of U using those vertices. This gives us a system $A\,(\mathbf{x}, \mathbf{t}) \le \bar{b}$ of polynomial length that describes U. ◀

Applying the above lemma for three polytopes R_1', R_2' and P with $n = 4$ and $r = 3$, we find a polytope $U \subset \mathbb{R}^{4+\ell}$ such that:

$$(x, \mathbf{y}, w) \in (R_1' \cup R_2' \cup P) \cap \mathbb{Z}^4 \quad \Longleftrightarrow \quad \exists \mathbf{t} \in \mathbb{Z}^\ell : (x, \mathbf{y}, w, \mathbf{t}) \in U \cap \mathbb{Z}^{4+\ell}. \tag{4.13}$$

Here we have $\ell = \lceil \log_2 3 \rceil = 2$, which means $\mathbf{t} \in \mathbb{Z}^2$ and $U \subset \mathbb{R}^6$. The lemma also allows us to find a system $A\,(x, \mathbf{y}, w, \mathbf{t}) \le \bar{b}$ that describes U, which has size polynomial in the systems in (4.11). Now, we can rewrite (4.10) as:

$$\exists x \in I \quad \forall \mathbf{y} \in J \quad \exists w \in \mathbb{Z} \quad : \quad \exists \mathbf{t} \in \mathbb{Z}^2 \quad (x, \mathbf{y}, w, \mathbf{t}) \in U,$$

which is equivalent to

$$\exists x \in I \quad \forall \mathbf{y} \in J \quad \exists \mathbf{z} \in \mathbb{Z}^3 \quad : \quad A\,(x, \mathbf{y}, \mathbf{z}) \le \bar{b}.$$

Here $\mathbf{z} = (w, \mathbf{t}) \in \mathbb{Z}^3$. The final system $A\,(x, \mathbf{y}, \mathbf{z}) \le \bar{b}$ still has size polynomial in the original input $\boldsymbol{\alpha}, N$ and ϵ. Therefore, the original GSA problem is equivalent to (4.1). This implies that (4.1) is NP-hard.

It remains to show that (4.1) is in NP. We argue that more general sentence (1.2) is also in NP. From a result in [14], if (1.2) is true, there must be an $\mathbf{x}$ satisfying it with length polynomial in the input P, A and $\bar{b}$. For such an $\mathbf{x}$, we can apply Theorem 1 to check the rest of the sentence, which has the form $\forall \mathbf{y} \exists \mathbf{z}$, in polynomial time. This shows that deciding (1.2) is in NP, and thus NP-complete. ◀

5 Proof of Theorem 4

Recall the definition of boxes from Section 4. In this section, we prove:

▶ **Theorem 13.** *Fix $k \geq 1$. Given a polytope $U \subset \mathbb{R}^{k+7}$ and finite boxes $I_1, \ldots, I_k \subset \mathbb{Z}$, $J \subset \mathbb{Z}^2$, $K \subset \mathbb{Z}^5$, then the problem of deciding:*

$$Q_1\, x_1 \in I_1 \quad \ldots \quad Q_k\, x_k \in I_k \quad \forall \mathbf{y} \in J \quad \exists \mathbf{z} \in K \quad : \quad (\mathbf{x}, \mathbf{y}, \mathbf{z}) \in U \tag{5.1}$$

is $\mathbf{\Sigma}_k^{\mathsf{P}}$ complete if $Q_1 = \exists$, and $\mathbf{\Pi}_k^{\mathsf{P}}$ complete if $Q_1 = \forall$. Here $Q_1, \ldots, Q_k \in \{\exists, \forall\}$ are k alternating quantifiers with $Q_k = \exists$. The polytope U is described by a system $A\,(\mathbf{x}, \mathbf{y}, \mathbf{z}) \leq \overline{b}$, where $A \in \mathbb{Z}^{m \times (k+7)}$ and $\overline{b} \in \mathbb{Z}^m$.

For the proof, we work with the canonical problem Q3SAT. Let Ψ a Boolean expression of the form:

$$\Psi(\mathbf{u}_1, \ldots, \mathbf{u}_k) = \bigwedge_{i=1}^{N} (a_i \vee b_i \vee c_i). \tag{5.2}$$

Here each $\mathbf{u}_j = (u_{j1}, \ldots, u_{j\ell}) \in \{0,1\}^\ell$ is a tuple of ℓ Boolean variables, and each a_i, b_i, c_i is a literal in the set $\{u_{js},\ \neg u_{js} \ : \ 1 \leq j \leq k,\ 1 \leq s \leq \ell\}$. From Ψ, we construct a sentence:

$$Q_1\, \mathbf{u}_1 \in \{0,1\}^\ell \quad Q_2\, \mathbf{u}_2 \in \{0,1\}^\ell \ \ldots\ Q_k\, \mathbf{u}_k \in \{0,1\}^\ell \quad : \quad \Psi(\mathbf{u}_1, \ldots, \mathbf{u}_k). \tag{5.3}$$

Here $Q_1, Q_2, \ldots, Q_k \in \{\forall, \exists\}$ are k alternating quantifiers with $Q_k = \exists$. The numbers ℓ and N are part of the input.

QUANTIFIED 3-SATISFIABILITY (Q3SAT)
Input: A Boolean expression Ψ of the form (5.2).
Decide: The truth of the sentence (5.3).

For clarity, we use the notation Q3SAT_k to emphasize problem (5.3) for a fixed k. It is well-known that Q3SAT_k is $\mathbf{\Sigma}_k^{\mathsf{P}}$-complete if $Q_1 = \exists$ and $\mathbf{\Pi}_k^{\mathsf{P}}$-complete if $Q_1 = \forall$ (see e.g. [23, 20] and [1]). We proceed to reduce (5.3) to (5.1). In fact, by representing each Boolean string $\mathbf{u}_j \in \{0,1\}^\ell$ as an integer $x_j \in [0, 2^\ell)$, we will only need to use $I_1 = I_2 = \cdots = I_k = [0, 2^\ell) \cap \mathbb{Z}$.

For every string $\mathbf{u}_j = (u_{j1}, \ldots, u_{j\ell}) \in \{0,1\}^\ell$, let $x_j \in [0, 2^\ell)$ be the corresponding integer in binary. Then u_{js} is true or false respectively when the s-th binary digit of x_j is 1 or 0. In other words, u_{js} is true or false respectively when $\lfloor x_j / 2^{s-1} \rfloor$ is odd or even. Observe that $t = \lfloor x_j / 2^{s-1} \rfloor$ is the only integer that satisfies $x_j/2^{s-1} - 1 < t \leq x_j/2^{s-1}$. Now, each term u_{js} or $\neg u_{js}$ can be expressed in x_j as follows:

$$\begin{aligned} u_{js} &\iff \exists w \in \mathbb{Z} \ : \ \begin{Bmatrix} 2w+1 & > & x_j/2^{s-1} - 1 \\ 2w+1 & \leq & x_j/2^{s-1} \end{Bmatrix}, \\ \neg u_{js} &\iff \exists w \in \mathbb{Z} \ : \ \begin{Bmatrix} 2w & > & x_j/2^{s-1} - 1 \\ 2w & \leq & x_j/2^{s-1} \end{Bmatrix}. \end{aligned} \tag{5.4}$$

Let $\mathbf{x} = (x_1, \ldots, x_k) \in [0, 2^\ell)^k$. Recall that each term a_i, b_i, c_i in (5.2) is u_{js} or $\neg u_{js}$ for some j and s. So each clause $a_i \vee b_i \vee c_i$ can be expressed in $\mathbf{x}$ as:

$$a_i \vee b_i \vee c_i \iff \exists w \in \mathbb{Z} \ : \ \left[D_i\,(\mathbf{x}, w) \leq \overline{d}_i\right] \vee \left[E_i(\mathbf{x}, w) \leq \overline{e}_i\right] \vee \left[F_i\,(\mathbf{x}, w) \leq \overline{f}_i\right], \tag{5.5}$$

where three systems $D_i(\mathbf{x}, w) \le \overline{d}_i$, $E_i(\mathbf{x}, w) \le \overline{e}_i$, $F_i(\mathbf{x}, w) \le \overline{f}_i$ are of the form (5.4) (with different j and s for each). Note that the strict inequalities in (5.4) can be sharpened without losing any integer solutions (see Remark 2). We define the polytopes:

$$\begin{aligned} K_i &= \big\{(\mathbf{x}, w) \in \mathbb{R}^{k+1} \ : \ x_1, \ldots, x_k, w \in [0, 2^\ell), \ D_i(\mathbf{x}, w) \le \overline{d}_i\big\},\\ L_i &= \big\{(\mathbf{x}, w) \in \mathbb{R}^{k+1} \ : \ x_1, \ldots, x_k, w \in [0, 2^\ell), \ E_i(\mathbf{x}, w) \le \overline{e}_i\big\},\\ M_i &= \big\{(\mathbf{x}, w) \in \mathbb{R}^{k+1} \ : \ x_1, \ldots, x_k, w \in [0, 2^\ell), \ F_i(\mathbf{x}, w) \le \overline{f}_i\big\}. \end{aligned}$$

So the RHS in (5.5) can be rewritten as:

$$\exists w \in \mathbb{Z} \ : \ (\mathbf{x}, w) \in K_i \cup L_i \cup M_i.$$

Let $I_1 = I_2 = \cdots = I_k = [0, 2^\ell) \cap \mathbb{Z}$, we see that (5.3) is equivalent to:

$$Q_1 x_1 \in I_1 \quad \ldots \quad Q_k x_k \in I_k \quad : \quad \bigwedge_{i=1}^{N} \Big[\exists w \in \mathbb{Z} \ : \ (\mathbf{x}, w) \in K_i \cup L_i \cup M_i\Big]. \tag{5.6}$$

For each i, we apply Lemma 12 (with $n = k+1$, $r = 3$) to the polytopes $K_i, L_i, M_i \subset \mathbb{R}^{k+1}$. This gives us another polytope $G_i \subset \mathbb{R}^{k+3}$ that satisfies:

$$(\mathbf{x}, w) \in K_i \cup L_i \cup M_i \quad \Longleftrightarrow \quad \exists \mathbf{v} \in \mathbb{Z}^2 \ : \ (\mathbf{x}, w, \mathbf{v}) \in G_i.$$

Substituting this into (5.6), we have an equivalent sentence:

$$Q_1 x_1 \in I_1 \quad \ldots \quad Q_k x_k \in I_k \quad : \quad \bigwedge_{i=1}^{N} \Big[\exists \mathbf{w} \in \mathbb{Z}^3 \ : \ (\mathbf{x}, \mathbf{w}) \in G_i\Big], \tag{5.7}$$

where $\mathbf{w} = (w, \mathbf{v}) \in \mathbb{Z}^3$, and each $G_i \subset \mathbb{R}^{k+3}$.

Notice that apart from the outer quantifiers, (5.7) is a direct analogue of (4.4), with G_i playing the role of P_i and $(\mathbf{x}, \mathbf{w})$ in place of (x, w). The proof now proceeds similarly to the rest of Section 4 after (4.4). Along the proof, we need to define G'_i and G in similar manners to (3.7) and (3.9). The variable $\mathbf{y} \in \mathbb{Z}^2$ is again needed to define G'_i. Φ and J from (3.2) are reused without change. This gives us $G'_i, G \subset \mathbb{R}^{k+5}$. At the end of the proof, we also need to apply Lemma 12 one more time to produce a single polytope U, just like in (4.13). The dimension 4 in (4.13) is now $k+5$. As a result, the final polytope U has dimension $k+7$. In the final form (5.1), we will have $\mathbf{x} \in \mathbb{Z}^k, \mathbf{y} \in \mathbb{Z}^2$ and $\mathbf{z} = (\mathbf{w}, \mathbf{t}) \in \mathbb{Z}^5$.

We have converted (5.3) to an equivalent sentence (5.1) with polynomial size. This shows that (5.1) is $\mathbf{\Sigma}_k^{\mathsf{P}}/\mathbf{\Pi}_k^{\mathsf{P}}$-hard depending when $Q_1 = \exists/\forall$. For each tuple $\mathbf{x} = (x_1, \ldots, x_k)$, we can check in polynomial time whether $\forall \mathbf{y} \in J \ \ \exists \mathbf{z} \in K \ : \ A(\mathbf{x}, \mathbf{y}, \mathbf{z}) \le \overline{b}$ by applying Theorem 1. This shows the membership of (5.1) in $\mathbf{\Sigma}_k^{\mathsf{P}}/\mathbf{\Pi}_k^{\mathsf{P}}$. We conclude that (5.1) is $\mathbf{\Sigma}_k^{\mathsf{P}}/\mathbf{\Pi}_k^{\mathsf{P}}$-complete when $Q_1 = \exists/\forall$. ◀

6 Proof of Theorem 7

6.1

Now we prove Theorem 7. We use the same construction as in the proof of Theorem 2. Recall the definition of $\{\{x\boldsymbol{\alpha}\}\}$ from Section 4. We reduce the following counting problem to a problem of the form (1.4):

#GOOD SIMULTANEOUS APPROXIMATIONS (#GSA)
Input: A rational vector $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_d) \in \mathbb{Q}^d$ and positive integers N, s_1, s_2.
Output: The number of integers $x \in [1, N]$ that satisfy $\{\{x\boldsymbol{\alpha}\}\} \leq s_1/s_2$.

The argument in [18] is based on a parsimonious reduction. Namely, it gives a bijection between solutions for #GSA and the following problem:

#WEAK PARTITIONS
Input: An integer vector $\overline{a} = (a_1, \ldots, a_d) \in \mathbb{Z}^d$.
Output: The number of $\mathbf{y} \in \{-1, 0, 1\}^d$ for which $\overline{a} \cdot \mathbf{y} = 0$.

It is well known and easy to see that #WEAK PARTITIONS is #P-complete. The decision version WEAK PARTITION was earlier shown by [27] to be NP-complete with a parsimonious reduction from KNAPSACK. Together with Lagarias's reduction, we conclude:

▶ **Theorem 14.** #GSA *is* #P*-complete.*

6.2

Now we proceed with the reduction of #GSA to (1.4).

Recall Φ and J from (3.2). We use the notations from section 3.1 and 3.2. Let P_i, P_i' and P be from (3.5), (3.7) and (3.9). Let Q_i, Q_i' and Q be from (3.6), (3.8) and (3.10). Let $I = [1, N] \cap \mathbb{Z}$. We have:

▶ **Observation 15.** *For every $x \in I$, there is a unique $w \in \mathbb{Z}$ such that $(x, w) \in Q_i$.*

Indeed, from (3.6), we have $(x, w) \in Q_i$ if and only if $x \in I$ and:

$$\alpha_i x + \epsilon - 1 < w \leq \alpha_i x + \epsilon.$$

For each $x \in I$, we get a half-open interval of length 1 for w, which has a unique integer.

▶ Remark 2. Note that each Q_i has an open edge defined by $\alpha_i x + \epsilon - 1 < w$. This can actually be sharpened without losing any integer point. Indeed, we can multiply the inequality with the denominators in α_i and ϵ, which have polynomial length. Then the resulting strict integer inequality of the form $a < b$ is equivalent to $a \leq b - 1$. Therefore, we can replace Q_i with a (smaller) closed parallelogram containing the same integer points. Taking the convex hull as in (3.10), we can similarly replace Q with a (smaller) closed polyhedron, without losing any integer points in Q.

▶ **Observation 16.** *For every $x \in I$ and $\phi_i \in \Phi$, there is a unique integer point $(x, \phi_i, w_i) \in Q$.*

Indeed, by (P11), for every $\phi_i \in \Phi$, we have $(x, \phi_i, w) \in Q$ if and only if $(x, w) \in Q_i$. Together with Observation 15, we have Observation 16.

Recall from (P5) that $P \subset Q$. Now consider the following set:

$$S = \big\{x \in I \ : \ \exists (\mathbf{y}, w) \in \Phi \times \mathbb{Z} \quad (x, \mathbf{y}, w) \in Q \setminus P\big\}.^2 \tag{6.1}$$

▶ **Lemma 17.** *For every $x \in I$, we have $\{\{x\boldsymbol{\alpha}\}\} > \epsilon$ if and only if $x \in S$.*

Proof. Assume $x \in S$, then there exist some $\phi_j \in \Phi$ and $w_j \in \mathbb{Z}$ so that $(x, \phi_i, w_i) \in Q \setminus P$. By Observation 16 and the fact that $P \subset Q$, there is no $w \in \mathbb{Z}$ for which $(x, \phi_i, w) \in P$. By (4.5), we have $\{\{x\boldsymbol{\alpha}\}\} > \epsilon$. Conversely, assume $\{\{x\boldsymbol{\alpha}\}\} > \epsilon$. By (4.5), there exist $\phi_i \in \Phi$ so that there is no $w \in \mathbb{Z}$ with $(x, \phi_i, w) \in P$. By Observation 16, the unique point (x, ϕ_i, w_i) in Q must be outside of P, i.e., $(x, \phi_i, w_i) \in Q \setminus P$. We conclude that $x \in S$ by (6.1). ◀

By the above lemma, counting S is equivalent to #GSA. The formulation (6.1) is very similar to (1.4), with $(\mathbf{y}, w)$ in place of $\mathbf{z}$. We cannot conclude directly that S is $\mathrm{E}_1(Q \setminus P)$ because of the restricted quantifier $\exists \mathbf{y} \in \Phi$ instead of $\exists \mathbf{y} \in \mathbb{Z}^2$. To turn S into the form (1.4), we need to convert $\exists \mathbf{y} \in \Phi$ to $\exists \mathbf{y} \in \mathbb{Z}^2$.

6.3

The final step is to modify the polytopes P and Q. In (6.1), we only consider projections of integer points $(x, \mathbf{y}, w) \in Q \setminus P$ with $\mathbf{y}$ restricted to the set Φ. In general, the complement $Q \setminus P$ has some other integer points $(x, \mathbf{y}, w)$ with $\mathbf{y}$ not lying in Φ. By (P12) such a point must necessarily have $\mathbf{y} \in R_2$. We can eliminate all of them by taking the convex hulls of P and Q with a "high enough" box over R_2. Below are the details.

Let $T = 1 + N \max_i \alpha_i$. By (3.5) and (3.6), we have $P_i, Q_i \subset [1, N] \times [-1, T]$. Recall from (3.7) and (3.9) that P is the convex hull of all P_i', which is simply P_i with an added second component ϕ_i. This leads to the following observation:

▶ **Observation 18.** *For every vector $\overline{\gamma} \in \mathbb{R}^2$, we have:*

$$\big\{(x, \mathbf{y}, w) \in P \ : \ \mathbf{y} = \overline{\gamma}\big\} \subseteq [1, N] \times \{\overline{\gamma}\} \times [-1, T].$$

The same holds for Q.

Next, consider the rectangular box J containing Φ and the complement $J \setminus \Phi$, where J is from (3.2). From properties (F3), (F4) and (F5), integer points in the complement $J \setminus \Phi$ lie in two separate convex polygons R_1 and R_2, as described in (3.3) and (3.4). We will only need R_2, which contains integer points below Φ. Define

$$R = \big\{(x, \mathbf{y}, w) \in \mathbb{R}^4 \quad : \quad x \in [1, N], \quad \mathbf{y} \in R_2, \quad w \in [-1, T]\big\}. \tag{6.2}$$

and

$$\widetilde{P} = \mathrm{conv}(P, R) \ , \quad \widetilde{Q} = \mathrm{conv}(Q, R) \quad \subset \quad \mathbb{R}^4. \tag{6.3}$$

For $\overline{\gamma} \in \mathbb{R}^2$, we denote by $P_{\overline{\gamma}}$ the set:

$$P_{\overline{\gamma}} = \big\{(x, \mathbf{y}, w) \in P \ : \ \mathbf{y} = \overline{\gamma}\big\},$$

and analogously for $\widetilde{P}_{\overline{\gamma}}$, $Q_{\overline{\gamma}}$, $\widetilde{Q}_{\overline{\gamma}}$ and $R_{\overline{\gamma}}$.

By Observation 18, for every $\overline{\gamma}$, we have $P_{\overline{\gamma}}, Q_{\overline{\gamma}} \subseteq [1, N] \times \{\overline{\gamma}\} \times [-1, T]$. From (6.2), we have $R_{\overline{\gamma}} = [1, N] \times \{\overline{\gamma}\} \times [-1, T]$ for every $\overline{\gamma} \in R_2$. Since $\widetilde{P} = \mathrm{conv}(P, R)$ and $\widetilde{Q} = \mathrm{conv}(Q, R)$, we have

$$\widetilde{P}_{\overline{\gamma}} = \widetilde{Q}_{\overline{\gamma}} = [1, N] \times \{\overline{\gamma}\} \times [-1, T] \quad \text{for every } \overline{\gamma} \in R_2. \tag{6.4}$$

For $\overline{\gamma} \in \Phi$, we claim that:

$$\widetilde{P}_{\overline{\gamma}} = P_{\overline{\gamma}} \quad \text{and} \quad \widetilde{Q}_{\overline{\gamma}} = Q_{\overline{\gamma}}. \tag{6.5}$$

Indeed, since $\overline{\gamma} \in \Phi$, we have $\overline{\gamma} = \phi_i$ and $P_{\overline{\gamma}} = P_{\phi_i}$ for some i. By (3.7) and (P8), we have $P_{\phi_i} = P_i'$. Since $R_2 \cap \Phi = \varnothing$, we have $\phi_i \notin R_2$. This implies $P_i' \cap R = \varnothing$, because R is a box over R_2, and P_i' is a parallelogram over ϕ_i. Recall from (P5) that P_i' forms a 2-dimensional face of P. Therefore, it still remains a 2-dimensional face of the convex hull $\widetilde{P} = \text{conv}(P, R)$. So $\widetilde{P}_{\overline{\gamma}} = P_{\overline{\gamma}} = P_i'$. The same argument applies to $\widetilde{Q}_{\overline{\gamma}}$ and $Q_{\overline{\gamma}}$.

Note that we also have $\widetilde{P} \subset \widetilde{Q}$, because $P \subset Q$. Consider the complement $\widetilde{Q} \setminus \widetilde{P}$. Assume $(x, \mathbf{y}, w) \in \mathbb{Z}^3$ is an integer point in $\widetilde{Q} \setminus \widetilde{P}$. By (6.4), such a point cannot exist for $\mathbf{y} \in R_2$. So we must have $\mathbf{y} \in \Phi$. Now by (6.5), we also have $(x, \mathbf{y}, w) \in Q \setminus P$. Therefore, from (6.1), we conclude that:

$$\begin{aligned} S &= \big\{x \in [1, N] \cap \mathbb{Z} \quad : \quad \exists (\mathbf{y}, w) \in \mathbb{Z}^3 \quad (x, \mathbf{y}, w) \in \widetilde{Q} \setminus \widetilde{P}\big\} \\ &= \big\{x \in [1, N] \cap \mathbb{Z} \quad : \quad \exists \mathbf{z} \in \mathbb{Z}^3 \quad (x, \mathbf{z}) \in \widetilde{Q} \setminus \widetilde{P}\big\}. \end{aligned}$$

Here $\mathbf{z} = (\mathbf{y}, w)$. The systems describing $\widetilde{Q}$ and $\widetilde{P}$ can be obtained in polynomial time from the input $\boldsymbol{\alpha}, N$ and ϵ. First, the vertices of P and Q are given by (P7) and (P10). The vertices of R directly come from those of R_2, which can be found from (3.4). By (6.3), we can obtain the vertices of $\widetilde{P}$ and $\widetilde{Q}$. The facets of $\widetilde{P}$ and $\widetilde{Q}$ can be found from their vertices in polynomial time, since both polytopes are in the fixed dimension 4. In summary, problem (1.4) applied to $\widetilde{P}$ and $\widetilde{Q}$ is #P-complete. This proves Theorem 7. ◀

6.4 Proof of Corollary 8

By Theorem 7, counting $|\mathrm{E}_1(Q \setminus P)|$ is #P-complete for $P \subset Q \subset \mathbb{R}^4$. Nevertheless, the complement $Q \setminus P$ can still be triangulated into polynomially many simplices $T_1 \sqcup \cdots \sqcup T_r$. In fact, by an application of Proposition 5.2.2 in [29], the systems describing all such T_i can be found in polynomial time. Therefore, counting $|\mathrm{E}_1(T_1 \sqcup \cdots \sqcup T_r)| = |\mathrm{E}_1(Q \setminus P)|$ is #P-complete. ◀

7 Final remarks and open problems

7.1

It is sufficient to prove Theorem 1 for the case when m, n are also bounded. In the system $A(\mathbf{x}, \mathbf{y}) \leq b$, we view $\mathbf{x}$ as the parameters and $\mathbf{y}$ as the variables to be solved for. For a fixed d_2 and $m \geq 2^{d_2}$, the *Doignon–Bell–Scarf theorem* [26, §16.5] implies that the system $A(\mathbf{x}, \mathbf{y}) \leq \overline{b}$ is solvable in $\mathbf{y} \in \mathbb{Z}^{d_2}$ if and only if every subsystem $A'(\mathbf{x}, \mathbf{y}) \leq \overline{b'}$ is solvable. Here A' is a submatrix with 2^{d_2} rows from A with $\overline{b'}$ the corresponding subvector from $\overline{b}$. In other words:

$$\exists \mathbf{y} \in \mathbb{Z}^{d_2} \quad A(\mathbf{x}, \mathbf{y}) \leq \overline{b} \quad \Longleftrightarrow \quad \bigwedge_{(A', \overline{b'})} \Big[\exists \mathbf{y} \in \mathbb{Z}^{d_2} \quad A'(\mathbf{x}, \mathbf{y}) \leq \overline{b'}\Big].$$

The total number of pairs $(A', \overline{b'})$ is $\binom{m}{2^{d_2}}$, which is polynomial in m.

Note that the conjunction over all $(A', \overline{b'})$ commutes with the universal quantifier $\forall \mathbf{x}$. Therefore:

$$\forall \mathbf{x} \in P \cap \mathbb{Z}^{d_1} \ \ \exists \mathbf{y} \in \mathbb{Z}^{d_2} \ \ A(\mathbf{x}, \mathbf{y}) \leq \overline{b} \iff \bigwedge_{(A', \overline{b'})} \Big[\forall \mathbf{x} \in P \cap \mathbb{Z}^{d_1} \ \ \exists \mathbf{y} \in \mathbb{Z}^{d_2} \ \ A'(\mathbf{x}, \mathbf{y}) \leq \overline{b'}\Big].$$

Thus, it is equivalent to check each of the smaller subproblems, each of which has $m = 2^{d_2}$. Recall that the number of facets in P is n, which can still be large. However, given the system $C\,\mathbf{x} \le \overline{\gamma}$ describing P, we can triangulate P into to a union of simplices $P_1 \sqcup \cdots \sqcup P_k$. Since the dimension d_1 is bounded, we can find such a triangulation in polynomial time (see e.g. [9]). Now for each pair $(A', \overline{b'})$, we have:

$$\forall \mathbf{x} \in P \cap \mathbb{Z}^{d_1} \;\; \exists \mathbf{y} \in \mathbb{Z}^{d_2} \;\; A'\,(\mathbf{x}, \mathbf{y}) \le \overline{b'} \iff \bigwedge_{i=1}^{k} \Big[\forall \mathbf{x} \in P_i \cap \mathbb{Z}^{d_1} \;\; \exists \mathbf{y} \in \mathbb{Z}^{d_2} \;\; A'\,(\mathbf{x}, \mathbf{y}) \le \overline{b'} \Big].$$

Each simplex $P_i \subset \mathbb{R}^{d_1}$ has $d_1 + 1$ facets. Each subsentence in the RHS now has $m = 2^{d_2}$ and $d_1 + 1$. Note that the total number of such subsentences is still polynomial, so it suffices to check each of them individually.

For three quantifiers $\exists \mathbf{x}\ \forall \mathbf{y}\ \exists \mathbf{z}$, this argument breaks down because the existential quantifier $\exists \mathbf{x}$ no longer commutes with a long conjunction.

7.2

By taking finite Boolean combinations, we see that Theorem 5 also allows counting integer points in a union of k polytopes, where k is bounded (see [3, 4]). In fact, Woods proved in [29, Prop. 5.3.1] that it is still possible to count all such points in polynomial time when k is arbitrary. By Corollary 8, we see that this is not the case for projection.

7.3

The GSA Problem plays an important role in both Number Theory and Integer Programming especially in connection to lattice reduction algorithms (see e.g. [15]). Let us mention that via a chain of parsimonious reductions one can show that #GSA is also hard to approximate (cf. [13]). Note also that GSA has been recently used in a somewhat related geometric context in [12].

7.4

An easy consequence of Lemma 12 proves the first part of the following result:

▶ **Proposition 19.** *Every set $S = \{\overline{p}_1, \ldots, \overline{p}_r\} \subset \mathbb{Z}^2$ is a projection of integer points of some convex polytope $P \subset \mathbb{R}^{2+d}$, where $d \le \lceil \log_2 r \rceil$. Moreover, the bound $d \le \lceil \log_2 r \rceil$ is tight.*

We only use the proposition to reduce the dimension of variable $\mathbf{z}$ in Theorem 9 from 4 to 3, but it is perhaps of independent interest. Note that a weaker inequality $d \le r$ is trivial.

Proof of the Second Part of Proposition 19. Consider a set $S = \{\overline{p}_1, \ldots, \overline{p}_r\}$ of integer points in convex position and with even coordinates. Assume there is a polytope $P \subset \mathbb{R}^{2+\ell}$ with $\ell < \lceil \log_2 r \rceil$ so that S is exactly the projection of $P \cap \mathbb{Z}^{2+\ell}$ on $\mathbb{Z}^2$. Then there are integer points $\overline{q}_1, \ldots, \overline{q}_r \in \mathbb{Z}^\ell$ so that $(\overline{p}_i, \overline{q}_i) \in P$. Since $r > 2^\ell$, by the pigeonhole principle, we have $\overline{q}_i - \overline{q}_j \in 2\mathbb{Z}^\ell$ for some $i \ne j$. Then the midpoint of $(\overline{p}_i, \overline{q}_i)$ and $(\overline{p}_j, \overline{q}_j)$ is an integer point in $\mathbb{Z}^{2+\ell}$, which also lies in P by convexity. The projection of this midpoint on $\mathbb{Z}^2$ is $(\overline{p}_i + \overline{p}_j)/2$, which must lie in S. However, the points in S are in convex positions and thus contain no midpoints, a contradiction. ◀

7.5

Let us give another motivation behind Theorem 7 and put it into context of our other work. In this paper, we bypass the "short generating function" technology developed for computing $|\mathrm{E}_1(P)|$ for convex polytopes $P \subset \mathbb{R}^d$. Note, however, that for $X = Q \setminus P$ as in the theorem, the corresponding short GF $f_X(\mathbf{t})$ is simply the difference $f_Q(\mathbf{t}) - f_P(\mathbf{t})$, which can still be computed in polynomial time (see [2]). Thus, if one could efficiently present the projection of $f_X(\mathbf{t})$ on $\mathbb{Z}$ as a short generating function of polynomial size, then one would be able to compute $|\mathrm{E}_1(Q \setminus P)|$, a contradiction. In other words, Theorem 7 is an extension of a result by Woods [28], which shows that computing projecting short generating functions is NP-hard. It is also an effective but weaker version of the main result in [21, Th. 1.3], which deals with the size of short GFs of the projections rather than complexity of their computation.

7.6

Corollary 8 says that computing $|\mathrm{E}_1(T_1 \cup \cdots \cup T_k)|$ is #P-complete even for simplices $T_i \subset \mathbb{R}^4$. By a stronger version of Theorem 6 (see [5]), for each polytope T_i, there is a short generating function $g_i(t)$ representing $\mathrm{E}_1(T_i)$. The union of all those generating functions correspond to $\mathrm{E}_1(T_1 \cup \cdots \cup T_k)$. As a corollary we conclude that the union operation on short generating functions is #P-hard to compute. As in §7.5 above, one should compare this to a stronger result [21, Th. 1.1], which says that the union of short generating functions can actually have super-polynomial lengths unless $\#\mathsf{P} \subseteq \mathsf{FP/poly}$.

7.7

It would be interesting to see if the dimension 4 in Theorem 7 is sharp and cannot be reduced to 3. One can argue both in favor and against this possibility. First, one can think of the result as a claim about complexity of nonconvex polyhedra $Q \setminus P$ in $\mathbb{R}^d$. For $d = 3$, the three dimensional nonconvex polyhedra are well known to be notoriously complicated to study via triangulations (see e.g. [24], the proof of the Th. 1.2 in [7] and a lengthly discussion in [9]). This suggests that for the "long" first coordinate dimensions of Q, it is unlikely that there is a good way to triangulate $Q \setminus P$ which would allow to compute $|\mathrm{E}_1(Q \setminus P)|$ efficiently.

To argue in the opposite direction, the problem of computing the number of integer points for polytopes in $\mathbb{R}^d$ becomes simpler for $d \leq 3$ (see e.g. [6, 8, 10]), so perhaps there is an ad hoc approach in this case.

7.8

Note that Theorem 9 was proved for dimensions $d_1 = 1, d_2 = 2$ and $d_3 = 3$. One can ask if the problem still remains NP-complete when some of these dimensions are lowered. In particular, it would be interesting to see if the following problem is still NP-complete:

$$\exists x \in P \cap \mathbb{Z} \quad \forall \mathbf{y} \in Q \cap \mathbb{Z}^2 \quad \exists \mathbf{z} \in \mathbb{Z}^2 \quad : \quad (x, \mathbf{y}, \mathbf{z}) \in U,$$

where $P \subset \mathbb{R}$, $Q \subset \mathbb{R}^2$ and $U \subset \mathbb{R}^5$ are convex polytopes.

Acknowledgements. We are grateful to Iskander Aliev, Matthias Aschenbrenner, Sasha Barvinok, Matt Beck, Artëm Chernikov, Jesús De Loera, Matthias Köppe, Sinai Robins and Kevin Woods for interesting conversations and helpful remarks.

References

1 S. Arora and B. Barak. *Computational complexity: A modern approach*. Cambridge University Press, Cambridge, 2009. doi:10.1017/CBO9780511804090.

2 A. Barvinok. A polynomial time algorithm for counting integral points in polyhedra when the dimension is fixed. In *34th Annual Symposium on Foundations of Computer Science (Palo Alto, CA, 1993)*, pages 566–572. IEEE Comput. Soc. Press, Los Alamitos, CA, 1993. doi:10.1109/SFCS.1993.366830.

3 A. Barvinok. *Integer points in polyhedra*. Zurich Lectures in Advanced Mathematics. European Mathematical Society (EMS), Zürich, 2008. doi:10.4171/052.

4 A. Barvinok and J. E. Pommersheim. An algorithmic theory of lattice points in polyhedra. In *New perspectives in algebraic combinatorics (Berkeley, CA, 1996–97)*, volume 38 of *Math. Sci. Res. Inst. Publ.*, pages 91–147. Cambridge Univ. Press, Cambridge, 1999.

5 A. Barvinok and K. Woods. Short rational generating functions for lattice point problems. *J. Amer. Math. Soc.*, 16(4):957–979, 2003. doi:10.1090/S0894-0347-03-00428-4.

6 M. Beck and S. Robins. *Computing the continuous discretely*. Undergraduate Texts in Mathematics. Springer, New York, second edition, 2015. Integer-point enumeration in polyhedra, With illustrations by David Austin. doi:10.1007/978-1-4939-2969-6.

7 A. Below, U. Brehm, J. A. De Loera, and J. Richter-Gebert. Minimal simplicial dissections and triangulations of convex 3-polytopes. *Discrete Comput. Geom.*, 24(1):35–48, 2000. doi:10.1007/s004540010058.

8 M. Brion. Points entiers dans les polytopes convexes. *Séminaire N. Bourbaki*, 36 (1993–1994):145–169, 1995. Talk No. 780. URL: http://www.numdam.org/item?id=SB_1993-1994__36__145_0.

9 J. A. De Loera, J. Rambau, and F. Santos. *Triangulations*, volume 25 of *Algorithms and Computation in Mathematics*. Springer-Verlag, Berlin, 2010. Structures for algorithms and applications. doi:10.1007/978-3-642-12971-1.

10 M. Dyer. On counting lattice points in polyhedra. *SIAM J. Comput.*, 20(4):695–707, 1991. doi:10.1137/0220044.

11 F. Eisenbrand. Integer programming and algorithmic geometry of numbers. In *50 years of integer programming 1958–2008*, pages xx+804. Springer-Verlag, Berlin, 2010. doi:10.1007/978-3-540-68279-0.

12 F. Eisenbrand and N. Hähnle. Minimizing the number of lattice points in a translated polygon. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1123–1130. SIAM, Philadelphia, PA, 2012.

13 F. Eisenbrand and T. Rothvoß. New hardness results for Diophantine approximation. In *Approximation, randomization, and combinatorial optimization*, volume 5687 of *Lecture Notes in Comput. Sci.*, pages 98–110. Springer, Berlin, 2009. doi:10.1007/978-3-642-03685-9_8.

14 E. Grädel. *The complexity of subclasses of logical theories*. PhD thesis, Universität Basel, 1978.

15 M. Grötschel, L. Lovász, and A. Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2 of *Algorithms and Combinatorics*. Springer-Verlag, Berlin, second edition, 1993. doi:10.1007/978-3-642-78240-4.

16 R. Kannan. Test sets for integer programs, ∀∃ sentences. In *Polyhedral combinatorics (Morristown, NJ, 1989)*, volume 1 of *DIMACS Ser. Discrete Math. Theoret. Comput. Sci.*, pages 39–47. Amer. Math. Soc., Providence, RI, 1990.

17 R. Kannan. Lattice translates of a polytope and the Frobenius problem. *Combinatorica*, 12(2):161–177, 1992. doi:10.1007/BF01204720.

18 J. Lagarias. The computational complexity of simultaneous Diophantine approximation problems. *SIAM J. Comput.*, 14(1):196–209, 1985. doi:10.1137/0214016.

19 H. Lenstra. Integer programming with a fixed number of variables. *Math. Oper. Res.*, 8(4):538–548, 1983. doi:10.1287/moor.8.4.538.

20 C. Moore and S. Mertens. *The nature of computation.* Oxford University Press, Oxford, 2011. doi:10.1093/acprof:oso/9780199233212.001.0001.

21 D. Nguyen and I. Pak. Complexity of short generating functions, preprint; arxiv:1702.08660, 2017.

22 D. Nguyen and I. Pak. Complexity of short presburger arithmetic. To appear in *STOC'17 – Proceedings of the 2017 ACM Symposium on Theory of Computing*, 2017.

23 C. H. Papadimitriou. *Computational complexity.* Addison-Wesley Publishing Company, Reading, MA, 1994.

24 J. Ruppert and R. Seidel. On the difficulty of triangulating three-dimensional nonconvex polyhedra. *Discrete Comput. Geom.*, 7(3):227–253, 1992. doi:10.1007/BF02187840.

25 U. Schöning. Complexity of Presburger arithmetic with fixed quantifier dimension. *Theory Comput. Syst.*, 30(4):423–428, 1997. doi:10.1007/s002240000059.

26 A. Schrijver. *Theory of linear and integer programming.* Wiley-Interscience Series in Discrete Mathematics. John Wiley & Sons, Ltd., Chichester, 1986. A Wiley-Interscience Publication.

27 P. van Emde Boas. Another NP-complete partition problem and the complexity of computing short vectors in a lattice. *Math. Dept. Report*, pages 81–04, 1981.

28 K. Woods. *Rational Generating Functions and Lattice Point Sets.* PhD thesis, University of Michigan, 2004.

29 K. Woods. Presburger arithmetic, rational generating functions, and quasi-polynomials. *J. Symb. Log.*, 80(2):433–449, 2015. doi:10.1017/jsl.2015.4.

On the Average-Case Complexity of MCSP and Its Variants*

Shuichi Hirahara[1] **and Rahul Santhanam**[2]

1 **Department of Computer Science, The University of Tokyo, Tokyo, Japan**
hirahara@is.s.u-tokyo.ac.jp
2 **Department of Computer Science, University of Oxford, Oxford, UK**
rahul.santhanam@cs.ox.ac.uk

Abstract

We prove various results on the complexity of MCSP (Minimum Circuit Size Problem) and the related MKTP (Minimum Kolmogorov Time-Bounded Complexity Problem):

- We observe that under standard cryptographic assumptions, MCSP has a *pseudorandom self-reduction*. This is a new notion we define by relaxing the notion of a random self-reduction to allow queries to be pseudorandom rather than uniformly random. As a consequence we derive a weak form of a worst-case to average-case reduction for (a promise version of) MCSP. Our result also distinguishes MCSP from natural NP-complete problems, which are not known to have worst-case to average-case reductions. Indeed, it is known that strong forms of worst-case to average-case reductions for NP-complete problems collapse the Polynomial Hierarchy.
- We prove the first non-trivial formula size lower bounds for MCSP by showing that MCSP requires nearly quadratic-size De Morgan formulas.
- We show average-case superpolynomial size lower bounds for MKTP against $\mathsf{AC}^0[p]$ for any prime p.
- We show the hardness of MKTP on average under assumptions that have been used in much recent work, such as Feige's assumptions, Alekhnovich's assumption and the Planted Clique conjecture. In addition, MCSP is hard under Alekhnovich's assumption. Using a version of Feige's assumption against co-nondeterministic algorithms that has been conjectured recently, we provide evidence for the first time that MKTP is not in coNP. Our results suggest that it might worthwhile to focus on the *average-case hardness* of MKTP and MCSP when approaching the question of whether these problems are NP-hard.

1998 ACM Subject Classification F.1.3 Complexity Measures and Classes

Keywords and phrases minimum circuit size problem, average-case complexity, circuit lower bounds, time-bounded Kolmogorov complexity, hardness

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.7

1 Introduction

Progress in complexity theory is often correlated with an improved understanding of *meta-computational problems*, i.e., problems that themselves encode questions about circuits or algorithms. Consider the canonical meta-computational problem Circuit-SAT, which asks whether a given circuit has a satisfying assignment. Results on the complexity of the

* The second author was supported by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013)/ERC Grant No. 615075. Part of this work was done during a visit of the first author to Oxford supported by the second author's ERC grant.

Circuit-SAT problem and its instantiations for restricted classes of circuits such as CNFs have had a major impact on complexity theory. These include the Cook-Levin theorem showing that CNF-SAT is NP-complete, the PCP theorem showing that CNF-SAT is hard to approximate, and more recently the connection established by Ryan Williams [35] between "non-trivial" algorithms for Circuit-SAT and circuit lower bounds, which has led to a new *algorithmic paradigm* for proving new circuit lower bounds.

A meta-computational problem that is in a sense dual to Circuit-SAT is MCSP: given the truth table of a Boolean function f and a parameter s, determine if f has circuits of size at most s. While Circuit-SAT asks about a property of the Boolean function encoded by a given circuit, MCSP asks about whether an explicitly given Boolean function can be encoded by a small circuit. It is easy to see that MCSP, like Circuit-SAT, is in NP; however, its precise complexity is much less well understood. This is despite the fact that MCSP was recognized as fundamental by theoretical computer scientists in the Soviet Union as early as the 1950s, as discussed in a fascinating survey by Trakhtenbrot [34]. In the more recent past, interest in MCSP was reawakened by a paper of Kabanets and Cai [24], building on the "Natural Proofs" work of Razborov and Rudich [30], and there have been several papers [5, 7, 10, 6, 9, 3, 27, 19, 18, 8] since on the complexity of the problem.

We do not have clear answers even to some basic questions about the complexity of MCSP. These questions include: Is MCSP NP-complete? Does the MCSP problem have similar structural properties to Circuit-SAT and other standard NP-complete problems, such as paddability and downward self-reducibility, or does it have properties such as random self-reducibility which are characteristic of problems such as Factoring and DiscreteLogarithm which are used in cryptography? What is the strongest evidence we can provide that MCSP is not in polynomial time? Are there formal connections between variants of MCSP which arise from using different circuit classes or fixing the parameter s in advance? Can we show unconditional complexity lower bounds for MCSP, for restricted classes of circuits such as sub-quadratic size formulas and sub-exponential size constant depth circuits with prime modular gates?

Our main argument in this paper is that it is valuable to look at MCSP through the lens of *average-case complexity*. By adopting this perspective, we are able to make progress on all of the questions above. We must first explain, however, what we mean by the average-case complexity of MCSP. Rather than studying MCSP itself, we study its parameterized version MCSP$[s]$, where the size function s is given in advance. We consider the complexity of this problem under the uniform distribution on inputs to the problem. Note that an input to the problem is simply the truth table of a Boolean function, so the distribution on inputs we consider corresponds to the uniform distribution on n-bit Boolean functions, which is fairly natural in this context. When the size function $s(n) = o(2^n/n)$, most Boolean functions do not have circuits of size s, and hence most inputs to the problem MCSP$[s]$ are NO inputs. Thus the problem is highly biased, and the algorithm just outputting NO on all instances has a very high probability of success. This means that it is uninteresting to consider a version of average-case complexity where the algorithm is allowed to make errors. Instead, we consider the standard zero-error notion, where the algorithm never outputs an incorrect answer, but is allowed to output '?' when it doesn't know the answer for an input.

Why do we believe studying the average-case complexity is more fruitful than studying the worst-case complexity? For one thing, it makes the theory cleaner. Let $\mathfrak{C}_1$ and $\mathfrak{C}_2$ be classes of circuits such that $\mathfrak{C}_1 \subseteq \mathfrak{C}_2$, and for a size function $s = o(2^n/n)$, let MCSP-$\mathfrak{C}_1[s]$ and MCSP-$\mathfrak{C}_2[s]$ be the variants of MCSP corresponding to the classes $\mathfrak{C}_1$ and $\mathfrak{C}_2$ respectively. Intuitively, it seems that MCSP-$\mathfrak{C}_2[s]$ must be at least as hard a problem as MCSP-$\mathfrak{C}_1[s]$

given that it concerns a more general class of circuits; however, in the setting of worst-case complexity, no formal connection between the two problems is known. In the setting of average-case complexity, in contrast, it is quite easy to show that the identity function is a reduction from the latter to the former. Similarly, given size functions s_1 and s_2 such that $s_1 \leq s_2$, it seems that $\mathsf{MCSP}[s_2]$ should be at least as hard as $\mathsf{MCSP}[s_1]$. Again, no formal reduction is known in the worst-case setting, but the identity function works as a reduction in the average-case setting. Thus the average-case setting seems to correspond more closely to our intuitions about the complexity of the problem.

A second point is that current techniques for proving results about the complexity of MCSP almost invariably yield results also on the average-case complexity. Most techniques we know go through some notion of pseudorandomness, and pseudorandomness is intrinsically an average-case notion. Thus, if we aim to prove results on the hardness of MCSP using current techniques, we should first aim for average-case hardness rather than worst-case hardness.

We now proceed to give a more detailed description of our results. In some cases, our results are not about MCSP but about a surrogate of it called MKTP. Rather than asking if the input has small circuits when interpreted as the truth table of a Boolean function, the MKTP problem asks if the input is compressible by a program from which individual bits of the input can be efficiently computed. Thus, while MCSP is a question about compression by circuits, MKTP is a question about compression by programs. For a formal definition, we refer to Section 2. Using known strong relationships between uniform and non-uniform complexity, MCSP and MKTP are closely related, and until recently all known results about one problem also applied to the other. A recent exception is [3], and our paper is another exception - it seems that hardness results are occasionally easier to show for MKTP because it corresponds to a more fine-grained notion of compressibility than MCSP.

1.1 Our Results

We first investigate the possibility of worst-case to average-case connections for MCSP. It is known that nonadaptive worst-case to average-case connections for NP-complete problems collapse the Polynomial Hierarchy [12]. Hence, if we could show a worst-case to average-case connection for MCSP, it would give strong evidence against the NP-hardness of MCSP. We are not able to do this; however, under standard cryptographic assumptions, we give a *pseudorandom self-reduction* for a promise version of MCSP. Recall that a random self-reduction is a reduction from a computational problem to itself where the queries are uniformly distributed and of the same length as the input. A random self-reduction gives a strong worst-case to average-case connection for a problem. Our notion of pseudorandom self-reduction relaxes the original notion by allowing the queries to be pseudorandomly distributed rather than randomly distributed. While our result does not give evidence that MCSP is not NP-complete, it does distinguish the MCSP problem from natural NP-complete problems, for which pseudorandom self-reductions are unknown even under standard cryptographic assumptions, to the best of our knowledge.

▶ **Theorem 1** (Pseudorandom self-reductions: Informal Version)**.** *Suppose exponentially hard one-way functions exist. Let s be a size function such that $s(n) = n^{\omega(1)}$ and $s(n) = o(2^n/n)$. There is a constant c such that there is a pseudorandom self-reduction for the promise version of* MCSP*, where the YES instances are truth tables of Boolean functions of circuit complexity at most $s(n) - n^c$ and the NO instances are truth tables of Boolean functions of circuit complexity at least $s(n) + n^c$.*

Though pseudorandom self-reductions do not give worst-case to average-case reductions in full generality as random self-reductions, they do give a weak version of such reductions, as described in more detail in Section 3. The proof idea we use to establish Theorem 1 is a twist on the idea used by [30] to rule out natural proofs under cryptographic assumptions.

Next we attempt to prove unconditional lower bounds for MCSP, and MKTP. We show such lower bounds in two settings where lower bounds were unknown. The first is a lower bound for MCSP against subquadratic De Morgan formulae, and the second is an average-case lower bound for MKTP against polynomial-size constant depth circuits with Mod p gates, for prime p. Both proofs exploit connections with pseudorandom generators, in the first case the pseudorandom generators of [20] against formulas, and in the second case the pseudorandom generators of [14] against $\mathsf{AC}^0[p]$ using resamplability. Note that in both settings traditional lower bound techniques such as the method of random restrictions, the Neciporuk technique and the polynomial method do not appear to be directly applicable.

▶ **Theorem 2** (Unconditional lower bounds: Informal Version)**.** *There are size functions s and s' such that* MCSP$[s]$ *does not have De Morgan formulae of size $N^{2-\Omega(1)}$, and* MKTP$[s']$ *cannot be decided with $\Omega(1)$ success on average by polynomial-size constant-depth circuits with Mod p gates, where p is any prime.*

Finally, and perhaps most strikingly, we show the hardness of MKTP on average under various assumptions that have been intensively studied recently, such as Feige's hypothesis [15], Alekhnovich's hypothesis [2] and the Planted Clique conjecture [23, 26]. These are the first hardness results for MCSP or MKTP under assumptions for problems that are not known to be in SZK. The fact that MKTP is hard on average under average-case hardness assumptions about NP-complete problems such as SAT and Clique might be taken as providing mild evidence in favour of the problem being NP-hard. Also, it has been conjectured by Ryan O'Donnell (personal communication) that Feige's hypothesis holds even with respect to co-nondeterministic polynomial time algorithms, and under this conjecture MKTP is not in coNP. We note that [5] observe that MKTP is not in coNP under a conjecture of Rudich [31], but the conjecture of O'Donnell is in our opinion more natural and plausible, relating as it does to Random Satisfiability, which is a problem that has been well studied algorithmically.

▶ **Theorem 3** (Hardness on Average under Popular Conjectures, Informal Version)**.** MKTP *is hard on average assuming Feige's hypothesis, Alekhnovich's hypothesis or the Planted Clique conjecture.* MCSP *is hard on average assuming Alekhnovich's hypothesis.*

2 Preliminaries and Notation

2.1 Boolean Function Complexity

We use $\mathcal{F}_m$ to denote the set of all Boolean functions $f\colon \{0,1\}^m \to \{0,1\}$. If W is a probability distribution, we use $w \sim W$ to denote an element sampled according to W. Similarly, for a finite set A, we use $a \sim A$ to denote that a is selected uniformly at random from A. Under this notation, $f \in \mathcal{F}_m$ represents a fixed function, while $f \sim \mathcal{F}_m$ is a uniformly random function. For convenience, we let $\mathcal{U}_n \stackrel{\text{def}}{=} \{0,1\}^n$. Following standard notation, $X \equiv Y$ denotes that random variables X and Y have the same distribution. We use standard asymptotic notation such as $o(\cdot)$ and $O(\cdot)$, and it always refer to a parameter $n \to \infty$, unless stated otherwise.

We say that $f, g \in \mathcal{F}_n$ are ε-close if $\Pr_{x\sim\mathcal{U}_n}[f(x) = g(x)] \geq 1 - \varepsilon$. We say that $h \in \mathcal{F}_n$ computes f with advantage δ if $\Pr_{x\sim\mathcal{U}_n}[f(x) = h(x)] \geq 1/2 + \delta$. It will sometimes be convenient to view a Boolean function $f \in \mathcal{F}_m$ as a subset of $\{0,1\}^m$ in the natural way.

We often represent Boolean functions as strings via the truth table mapping. Given a Boolean function $f \in \mathcal{F}_n$, $\mathtt{tt}(f)$ is the 2^n-bit string which represents the truth table of f in the standard way, and conversely, given a string $y \in \{0,1\}^{2^n}$, $\mathtt{fn}(y)$ is the Boolean function in $\mathcal{F}_n$ whose truth table is represented by y.

We identify each Boolean function $f : \{0,1\}^* \to \{0,1\}$ with a language $L \subseteq \{0,1\}^*$, where for any $x \in \{0,1\}^*$, $x \in L$ iff $f(x) = 1$. A promise problem is a pair (Π_{YES}, Π_{NO}) of languages over $\{0,1\}$, such that $\Pi_{YES} \cap \Pi_{NO} = \emptyset$. We say a language $L \subseteq \{0,1\}^*$ is consistent with a promise problem (Π_{YES}, Π_{NO}) if $\Pi_{YES} \subseteq L$ and $\Pi_{NO} \subseteq \bar{L}$.

Let $\mathfrak{C} = \{\mathcal{C}_n\}_{n\in\mathbb{N}}$ be a class of Boolean functions, where each $\mathcal{C}_n \subseteq \mathcal{F}_n$. Given a language $L \subseteq \{0,1\}^*$, we write $L \in \mathfrak{C}$ if for every large enough n we have that $L_n \stackrel{\text{def}}{=} \{0,1\}^n \cap L$ is in $\mathcal{C}_n$. Often we will abuse notation and view $\mathfrak{C}$ as a class of Boolean circuits. For convenience, we use number of wires to measure circuit size. We denote by $\mathfrak{C}[s(n)]$ the set of n-variable $\mathfrak{C}$-circuits of size at most $s(n)$. As usual, we say that a uniform complexity class Γ is contained in $\mathfrak{C}[\mathsf{poly}(n)]$ if for every $L \in \Gamma$ there exists $k \geq 1$ such that $L \in \mathfrak{C}[n^k]$.

Given a sequence of Boolean functions $\{f_n\}_{n\in\mathbb{N}}$ with $f_n \colon \{0,1\}^n \to \{0,1\}$, we let $\mathfrak{C}^f$ denote the extension of $\mathfrak{C}$ that allows $\mathcal{C}_n$-circuits to have oracle gates computing f_n.

We will use a few other standard notions, and we refer to standard textbooks in computational complexity and circuit complexity for more details.

2.2 Natural Proofs and the Minimum Circuit Size Problem

We say that $\mathfrak{R} = \{\mathcal{R}_n\}_{n\in\mathbb{N}}$ is a *combinatorial property* (of Boolean functions) if $\mathcal{R}_n \subseteq \mathcal{F}_n$ for all n. We use $L_{\mathfrak{R}}$ to denote the language of truth-tables of functions in $\mathfrak{R}$. Formally, $L_{\mathfrak{R}} = \{y \mid y = \mathtt{tt}(f) \text{ for some } f \in \mathcal{R}_n \text{ and } n \in \mathbb{N}\}$.

▶ **Definition 4** (Natural Properties [30]). Let $\mathfrak{R} = \{\mathcal{R}_n\}$ be a combinatorial property, $\mathfrak{C}$ a circuit class, and $\mathfrak{D}$ a (uniform or non-uniform) complexity class. We say that $\mathfrak{R}$ is a $\mathfrak{D}$-natural property useful against $\mathfrak{C}[s(n)]$ if there is $n_0 \in \mathbb{N}$ such that the following holds:

(i) *Constructivity.* $L_{\mathfrak{R}} \in \mathfrak{D}$.
(ii) *Density.* For every $n \geq n_0$, $\Pr_{f\sim\mathcal{F}_n}[f \in \mathcal{R}_n] \geq 1/2^{O(n)}$.
(iii) *Usefulness.* For every $n \geq n_0$, we have $\mathcal{R}_n \cap \mathcal{C}_n[s(n)] = \emptyset$.

▶ **Definition 5** (Minimum Circuit Size Problem). Let $\mathfrak{C}$ be a circuit class. The Minimum Circuit Size Problem for $\mathfrak{C}$, abbreviated as MCSP-$\mathfrak{C}$, is defined as follows:

- *Input.* A pair (y, s), where $y \in \{0,1\}^{2^n}$ for some $n \in \mathbb{N}$, and $1 \leq s \leq 2^n$ is an integer (inputs not of this form are rejected).
- *Question.* Does $\mathtt{fn}(y)$ have $\mathfrak{C}$-circuits of size at most s?

We also define variants of this problem, where the circuit size is not part of the input.

▶ **Definition 6** (Parameterized Minimum Circuit Size Problem). Let $\mathfrak{C}$ be a circuit class, and $s \colon \mathbb{N} \to \mathbb{N}$ be a function. The Minimum Circuit Size Problem for $\mathfrak{C}$ with parameter s, abbreviated as MCSP-$\mathfrak{C}[s]$, is defined as follows:

- *Input.* A string $y \in \{0,1\}^{2^n}$, where $n \in \mathbb{N}$ (inputs not of this form are rejected).
- *Question.* Does $\mathtt{fn}(y)$ have $\mathfrak{C}$-circuits of size at most $s(n)$?

▶ **Definition 7** (Parameterized Minimum Circuit Size Gap Problem). Let $\mathfrak{C}$ be a circuit class, and let $c, s \colon \mathbb{N} \to \mathbb{N}$ be functions such that $c(n) \geq s(n)$ for all $n \in \mathbb{N}$. The Minimum Circuit Size Gap Problem for $\mathfrak{C}$ with parameters c and s, abbreviated as MCSP-$\mathfrak{C}[c, s]$ is a promise problem defined as follows:

- *YES Instance.* Any string $y \in \{0,1\}^{2^n}$, where $n \in \mathbb{N}$, such that $\mathtt{fn}(y)$ has $\mathfrak{C}$-circuits of size at most $s(n)$.
- *NO Instance.* Any string $y \in \{0,1\}^{2^n}$, where $n \in \mathbb{N}$, such that $\mathtt{fn}(y)$ has no $\mathfrak{C}$-circuits of size at most $c(n)$.

When $\mathfrak{C}$ is not explicitly specified, we take $\mathfrak{C}$ to be the class of Boolean circuits.

Note that a dense property useful against $\mathfrak{C}[s(n)]$ is a dense subset of the complement of MCSP-$\mathfrak{C}[s]$.

2.3 Time-Bounded Kolmogorov Complexity and MKTP

KT-complexity was proposed in [5] in order to model circuit complexity in terms of time-bounded Kolmogorov complexity: it is known that $\mathrm{KT}(\mathtt{tt}(f))$ and the minimum circuit size of f are polynomially-related to each other. As usual, we fix a universal random-access Turing machine U that simulates all Turing machines efficiently. The KT-complexity of a string x is the minimum of $|d| + t$, where d is a string for describing x implicitly and t is the time it takes to output x. More formally, we have the definition below, where U^d denotes the Turing machine U with random access to the string d:

▶ **Definition 8** (KT-complexity [5])**.** Let $x = x_1 \cdots x_n \in \{0,1\}^n$. The KT-complexity of x is defined as follows.

$$\mathrm{KT}(x) := \min\{|d| + t \mid \ U^d(i) = x_i \text{ in } t \text{ steps for any } 1 \leq i \leq n+1 \ \}.$$

Here, x_{n+1} is defined as $\perp$ (a stop symbol).

For this complexity measure, a problem related to MCSP is defined as follows.

▶ **Definition 9** (Minimum Kolmogorov Time-bounded Complexity Problem)**.** The Minimum Kolmogorov Time-bounded Complexity Problem, abbreviated as MKTP, is defined as follows:

- *Input.* A pair (y, s), where $y \in \{0,1\}^*$ and $s \in \mathbb{N}$ (inputs not of this form are rejected).
- *Question.* $\mathrm{KT}(y) \leq s$?

▶ **Definition 10** (Parameterized Minimum Kolmogorov Time-bounded Complexity Problem)**.** Let $s\colon \mathbb{N} \to \mathbb{N}$ be a function. The Minimum Kolmogorov Time-bounded Complexity Problem with parameter s, abbreviated as MKTP$[s]$, is defined as follows:

- *Input.* A string $y \in \{0,1\}^*$.
- *Question.* $\mathrm{KT}(y) \leq s(|y|)$?

2.4 Average-Case Complexity

We require various notions of easiness on average. Let $\mathcal{D} = \{D_n\}, n \in \mathbb{N}$, be a sequence of distributions, where each D_n has support in $\{0,1\}^n$. A *distributional problem* is a pair $(L, \mathcal{D})$, where $L \subseteq \{0,1\}^*$ and $\mathcal{D}$ is a sequence of distributions.

▶ **Definition 11** (Easiness on Average)**.** Let $\mathfrak{C}$ be a (uniform or non-uniform) complexity class, and let $\varepsilon : \mathbb{N} \to [0,1]$ be a success parameter. We say a distributional problem $(L, \mathcal{D})$ is solvable in $\mathfrak{C}$ on average with success ε if there is a $\mathfrak{C}$-algorithm A such that for each $x \in \{0,1\}^*$, $A(x) = L(x)$ or $A(x) =' ?'$, and for each $n \in \mathbb{N}$, with probability at least $\varepsilon(n)$ over $x \sim D_n$, $A(x) = L(x)$. We say that a language L is in $\mathfrak{C}$ on average with success ε if (L, U_n) is in $\mathfrak{C}$ on average with success ε.

We observe that the easiness on average of certain variants of MCSP is equivalent to natural proofs, and moreover that the easiness on average is robust with regard to the success parameter.

▶ **Proposition 12** (Natural Proofs and Easiness of MCSP on Average). *Let $s : \mathbb{N} \to \mathbb{N}$ be a size function such that $s(n) = 2^n/n^{\omega(1)}$. The following are equivalent:*

1. *For all $c > 0$ there are* P*-natural (resp.* SIZE(poly)*-natural) properties useful against* SIZE$(s(cn))$.
2. *For all $c > 0$* MCSP$[s(cn)]$ *is solvable in* P *(resp.* SIZE(poly)*) on average with success* $1/\operatorname{poly}(N)$*, where $N = 2^n$ is the input size for the* MCSP *problem.*
3. *For all $c > 0$* MCSP$[s(cn)]$ *is solvable in* P *(resp.* SIZE(poly)*) on average with success probability* $1 - 1/\operatorname{poly}(N)$.

Proof. We provide just a sketch. Let $\mathfrak{C}$ be P or SIZE(poly). The proof is based on two observations. The first is that a $\mathfrak{C}$-natural property of density ε useful against $s(n)$-size Boolean circuits immediately yields that MCSP$[s]$ is in $\mathfrak{C}$ on average with success ε, simply by using the constructivity of the property and answering '?' on any input truth table that does not satisfy the property. Conversely, if MCSP$[s]$ is in $\mathfrak{C}$ on average with success ε, this implies a $\mathfrak{C}$-natural property with density $\varepsilon - 1/N^{\omega(1)}$, where a truth table is in the property iff the average-case algorithm answers 0 on the truth table. Since $s(n) = 2^n/n^{\omega(1)}$, the algorithm can only answer 1 on a $1/N^{\omega(1)}$ fraction of inputs, and hence the density of inputs on which the algorithm answers 0 is at least $\varepsilon - 1/N^{\omega(1)}$.

The second observation is that the density for natural properties can be amplified, with some cost to the usefulness. Given a natural property $\mathfrak{R}$, we define a property $\mathfrak{R}'$ by splitting up the input truth table for $\mathfrak{R}'$ into independent blocks, and accepting iff at least one of the blocks satisfies $\mathfrak{R}$. A simple calculation shows that by choosing the block size appropriately, a property with density $1/\operatorname{poly}(N)$ can be transformed into one with density $1 - 1/\operatorname{poly}(N)$, with the new property being useful against circuits of size $s(cn)$ for some $c > 0$ if the original property is useful against circuits of size $s(n)$. ◀

We also introduce and use a more refined definition of easiness on average, where we separate the complexity of the algorithm solving the problem on average from the complexity of the error set (or more precisely a not too large superset of the error set).

▶ **Definition 13** (Easiness on Average with Bounded Complexity Error Set). Let $\mathfrak{B}$ and $\mathfrak{C}$ be (uniform or non-uniform) complexity classes, and let $\varepsilon : \mathbb{N} \to [0, 1]$ be a success parameter. We say a distributional problem $(L, \mathcal{D})$ is solvable in $\mathfrak{C}$ on average with $\mathfrak{B}$-bounded success ε if there is a $\mathfrak{C}$-algorithm A and a $\mathfrak{B}$-algorithm A' such that for each $x \in \{0,1\}^*$, $A(x) = L(x)$ or $A(x) =' ?'$, $A(x) =' ?'$ implies $A'(x) = 1$, and for each $n \in \mathbb{N}$, with probability at least $\varepsilon(n)$ over $x \sim D_n$, $A'(x) = 0$. We say that a language L is in $\mathfrak{C}$ on average with $\mathfrak{B}$-bounded success ε if (L, U_n) is in $\mathfrak{C}$ on average with $\mathfrak{B}$-bounded success ε. We say that a language L is feasibly in $\mathfrak{C}$ on average with success ε if L is in $\mathfrak{C}$ on average with SIZE(poly)-bounded success ε.

We observe that the above notion is equivalent to the notion in Definition 11 when $\mathfrak{B} = \mathfrak{C}$ and $\mathfrak{C}$ is a standard complexity class such as P or SIZE(poly).

▶ **Proposition 14** (Specialization of the Refined Notion of Average-Case Easiness to the Standard Notion). *Let $\mathfrak{C} =$* P *or* SIZE(poly)*, and $\varepsilon : \mathbb{N} \to [0, 1]$ be a success parameter. L is in $\mathfrak{C}$ on average with $\mathfrak{C}$-bounded success ε iff L is in $\mathfrak{C}$ on average with success ε.*

Proof. The forward direction is immediate. For the backward direction, let A be a $\mathfrak{C}$-algorithm solving L on average with success ε. We define a $\mathfrak{C}$-algorithm A' as follows: $A'(x) = 1$ iff $A(x) =' ?'$. It is easy to see that A and A' satisfy the conditions in Definition 13, showing that L is in $\mathfrak{C}$ on average with $\mathfrak{C}$-bounded success ε. ◀

2.5 Randomness and Pseudorandomness

▶ **Definition 15** (Indistinguishability)**.** Let $\mathfrak{C}$ be a (uniform or non-uniform) complexity class, and $\{D_n\}, \{D'_n\}, n \in \mathbb{N}$ be sequences of distributions such that for each n, D_n and D'_n are supported on $\{0,1\}^n$. Let $\varepsilon : \mathbb{N} \to [0,1]$ be an error parameter. We say $\{D_n\}$ and $\{D'_n\}$ are ε-indistinguishable by $\mathfrak{C}$ if for all $L \in \mathfrak{C}$ and all large enough n,

$$\left| \Pr_{w \sim D_n}[L(w) = 1] - \Pr_{w \sim D'_n}[L(w) = 1] \right| \leq \varepsilon(n).$$

By default, the parameter $\varepsilon(n)$ in the above definition is taken to be $1/n$.

▶ **Definition 16** (Pseudorandom Generators)**.** Let $\ell \colon \mathbb{N} \to \mathbb{N}$, $h \colon \mathbb{N} \to \mathbb{N}$ and $\varepsilon \colon \mathbb{N} \to [0,1]$ be functions, and let $\mathfrak{C}$ be a circuit class. A sequence $\{G_n\}$ of functions $G_n \colon \{0,1\}^{\ell(n)} \to \{0,1\}^n$ is an (ℓ, ε) pseudorandom generator *(PRG)* against $\mathfrak{C}$ if $\{G_n(U_{\ell(n)})\}$ and $\{U_n\}$ are ε-indistinguishable by $\mathfrak{C}$. The pseudorandom generator is called quick if its range is computable in time $2^{O(\ell(n))}$.

We define random reducibility between languages, and random self-reducibility.

▶ **Definition 17** (Random Self-Reducibility)**.** Let $L, L' \subseteq \{0,1\}^*$ be languages. L is said to be randomly reducible to L' if there are constants k, ℓ and polynomial-time computable functions $g \colon \{0,1\}^* \to \{0,1\}^*$ and $h \colon \{0,1\}^* \to \{0,1\}$ satisfying the following conditions:

1. For large enough n, for every $x \in \{0,1\}^n$ and for each $i \in \mathbb{N}$ such that $1 \leq i \leq n^k$, $g(i, x, r) \sim U_n$ when $r \sim U_{n^\ell}$.
2. For large enough n and for every $x \in \{0,1\}^n$:

$$L(x) = h(x, r, L'(g(1, x, r)), L'(g(2, x, r)), \ldots, L'(g(n^k, x, r)))$$

with probability $\geq 1 - 2^{-n}$ when $r \sim U_{n^\ell}$.

We say L is randomly self-reducible if L is randomly reducible to L. Also, given a promise problem $Q = (\Pi_{YES}, \Pi_{NO})$ and a language L, we say Q is randomly reducible to L if the first condition above holds for all large enough strings but the second condition is only required to hold for strings $x \in \Pi_{YES} \cup \Pi_{NO}$.

We also define a new notion of *pseudorandom reducibility* by relaxing the first condition in the above definition.

▶ **Definition 18** (Pseudorandom Self-Reducibility)**.** Let $\mathfrak{C}$ be a complexity class. Let $Q = (\Pi_{YES}, \Pi_{NO})$ be a promise problem, where $\Pi_{YES}, \Pi_{NO} \subseteq \{0,1\}^*$, and let $L \subseteq \{0,1\}^*$ be a language. Q is said to be pseudorandomly reducible to L with respect to $\mathfrak{C}$ if there are constants k, ℓ and polynomial-time computable functions $g \colon \{0,1\}^* \to \{0,1\}^*$ and $h \colon \{0,1\}^* \to \{0,1\}$ satisfying the following conditions:

1. For every sequence $\{(x_n, i_n)\}, n \in \mathbb{N}$ where $x_n \in \{0,1\}^n$ and $1 \leq i_n \leq n^k$ for all $n \in \mathbb{N}$, $\{g(i_n, x_n, U_{n^\ell})\}$ and $\{U_n\}$ are indistinguishable by $\mathfrak{C}$.

2. For large enough n and for every $x \in (\Pi_{YES} \cup \Pi_{NO}) \cap \{0,1\}^n$:

$$L(x) = h(x, r, L(g(1, x, r)), L(g(2, x, r)), \ldots, L(g(n^k, x, r)))$$

with probability $\geq 1 - 2^{-n}$ when $r \sim U_{n^\ell}$.

Q is said to be pseudorandomly self-reducible with respect to $\mathfrak{C}$ if there is a language L consistent with Q such that Q is pseudorandomly reducible to L with respect to $\mathfrak{C}$.

▶ **Definition 19** (Pseudorandom Functions)**.** Let $s : \mathbb{N} \to \mathbb{N}$ be a size function, and let $\mathfrak{C}$ be a complexity class. A pseudo-random function generator (PRFG) with seed length ℓ against $\mathfrak{C}$ is a sequence of functions $\{F_n\}, F_n : \{0,1\}^{\ell(n)} \to \{0,1\}^{2^n}$ such that the function $G_n(z, i)$ giving the i'th bit of $F_n(z)$ is computable in time $\mathrm{poly}(n)$, and the distributions $F_n(U_{\ell(n)})$ and U_{2^n} are $1/2^n$ -indistinguishable by $\mathfrak{C}$.

We note that the definition of pseudorandom functions given here is somewhat different from the standard notion [16], where the distinguisher circuit only gets oracle access to the function it is trying to distinguish from random. Our notion is stronger, and more relevant to the current setting. As shown by [30], the construction of [16] gives pseudorandom functions according to Definition 19, under the assumption that exponentially hard one-way functions exist. We now define this concept.

▶ **Definition 20** (Exponentially Hard One-way Functions)**.** A sequence $\{f_n\}, n \in \mathbb{N}$ of functions, where $f_n : \{0,1\}^n \to \{0,1\}^n$ is said to be a exponentially hard one-way function if $\{f_n\}$ is polynomial-time computable, and there is a constant $\varepsilon > 0$ such that for any sequence $\{C_n\}$ of circuits, where C_n has size at most 2^{n^ε} for large enough n, $\Pr_{y \sim U_n}(f_n(C_n(f_n(y))) = f_n(y)) < 1/2^{n^\varepsilon}$.

▶ **Theorem 21** (PRFG from One-Way Functions; Goldwasser-Goldreich-Micali [16])**.** *If exponentially hard one-way functions exist, then there is a PRFG with seed length* $\mathrm{poly}(n)$ *against* $\mathsf{SIZE}(\mathrm{poly})$.

In a seminal result with significant implications for the provability of circuit lower bounds, Razborov and Rudich [30] showed that the existence of pseudorandom functions is incompatible with the existence of natural properties.

▶ **Theorem 22** (Ruling out Natural Properties using Pseudorandom Function [30])**.** *If exponentially hard one-way functions exist, then there are no* $\mathsf{SIZE}(\mathrm{poly})$*-natural properties useful against* $\mathsf{SIZE}(\mathrm{poly})$.

3 Pseudorandom Self-Reducibility for MCSP

▶ **Theorem 23.** *Suppose exponentially hard one-way functions exist. Let* $s : \mathbb{N} \to \mathbb{N}$ *be a size bound such that* $s(n) = n^{\omega(1)}$*. Then there is a constant* $c > 0$ *such that* $\mathsf{MCSP}[s + n^c, s - n^c]$ *is pseudorandomly self-reducible with respect to* $\mathsf{SIZE}(\mathrm{poly})$.

Proof. Suppose that exponentially hard one-way functions exist. By Theorem 21, there is a PRFG $\{F_n\}$ with seed length $\mathrm{poly}(n)$ against $\mathsf{SIZE}(\mathrm{poly})$. Let c be a constant such that the function G_n corresponding to F_n in Definition 19 is computable in time n^d for some constant $d < c$, and hence by Boolean circuits of size $< n^c$, using the standard simulation of deterministic time by circuit size. We show that there is a pseudorandom reduction from $\mathsf{MCSP}[s + n^c, s - n^c]$ to $\mathsf{MCSP}[s]$. As the language $\mathsf{MCSP}[s]$ is consistent

with the promise problem $\mathsf{MCSP}[s + n^c, s - n^c]$, this yields a pseudorandom self-reduction for $\mathsf{MCSP}[s + n^c, s - n^c]$.

The idea is that the pseudorandom self-reduction is a 1-query reduction which uses its randomness to generate a function pseudorandomly and then XORs the pseudorandom function with the input truth table. It is not hard to see that the output of this process is still pseudorandom; however, the circuit size of the Boolean function corresponding to the output differs from the circuit size of the function corresponding to the input by at most n^c.

We define functions g and h which define a pseudorandom reduction by Definition 18. We choose the constant k to be 0, i.e., this is a pseudorandom reduction which makes just 1 query. Hence we can assume that g has just 2 parameters y and r. Let $y \in \{0,1\}^N$ be an input, where $N = 2^n$. This is the only case we need to argue about – when N is not a power of 2, we can define $g(y, r)$ to be a uniformly random string of length N, and $h(y, r, b) = b$ for any y of length N, random string r and bit b; then, the conditions of Definition 18 are satisfied, as no strings of length N are YES instances of either the promise problem $\mathsf{MCSP}[s + n^c, s - n^c]$ or the language $\mathsf{MCSP}[s]$. In what follows, we assume $N = 2^n$.

The pseudorandom reduction uses a random string r of length $\ell(n)$, where ℓ is the seed length for the PRFG given by Theorem 21 against $\mathsf{SIZE}(\text{poly})$. We define $g(y, r) = y \oplus F_n(r)$. As in the previous paragraph, we define $h(y, r, b) = b$ for any y of length N, random string r of length $\ell(n)$ and bit b.

We argue that this is indeed a valid pseudorandom reduction from $\mathsf{MCSP}[s+n^c, s-n^c]$ to $\mathsf{MCSP}[s]$ for any size function $s(n) = n^{\omega(1)}$. First we need to show that for any sequence $\{y_N\}$ of inputs, where $|y_N| = N$, $g(y_N, U_{\ell(n)})$ and U_N are $1/N$-indistinguishable by $\mathsf{SIZE}(\text{poly})$. Suppose, to the contrary, that there is a sequence of circuits $\{C_N\}$ $1/N$-distinguishing the two distributions, where the size of C_N is $\text{poly}(n)$. Consider the sequence of circuits $\{C'_N\}$, where $C'_N(z) = C_N(z \oplus y_N)$ for any input z of length N. The circuits $\{C'_N\}$ are also of size $\text{poly}(N)$, and it is easy to see that they $1/2^n$-distinguish the distributions $F_n(U_{\ell(n)})$ and U_N, using the fact that $N = 2^n$. But this is a contradiction to the assumption that $\{F_n\}$ is a PRFG against $\mathsf{SIZE}(\text{poly})$.

Next, we need to show that for any y of length N, if y is a YES instance of $\mathsf{MCSP}[s+n^c, s-n^c]$, then $h(y, r, \mathsf{MCSP}[s](g(y, r))) = 1$ with probability at least $1 - 2^{-n}$ over the choice of r, and similarly, if y is a NO instance of $\mathsf{MCSP}[s+n^c, s-n^c]$, then $h(y, r, \mathsf{MCSP}[s](g(y, r))) = 0$ with probability at least $1-2^{-n}$ over the choice of r. In the former case, we have that $\mathtt{fn}(y)$ has circuit complexity at most $s-n^c$, as y is a YES instance. Hence for any r, $f' = \mathtt{fn}(y \oplus F_n(r))$ has circuit complexity at most s, since the function G_n corresponding to F_n is computable by circuits of size less than n^c by assumption, for any r. Therefore $\mathsf{MCSP}[s](g(y, r) = \mathsf{MCSP}[s](\mathtt{tt}(f')) = 1$ with probability 1 over r, and hence $h(y, r, \mathsf{MCSP}[s](g(y, r))) = 1$ with probability 1 over r, as h just outputs its last parameter. A completely analogous argument establishes the claim for an arbitrary NO instance. ◀

▶ **Theorem 24.** *Let $\mathfrak{B}$ and $\mathfrak{C}$ be complexity classes such that $\mathfrak{C}$ contains BPP and is closed under probabilistic polynomial-time disjunctive truth-table reductions, and let $s : \mathbb{N} \to \mathbb{N}$ be a size function. Let $\varepsilon : \mathbb{N} \to [0, 1]$ be a success parameter such that $\varepsilon \geq 2/N$. Suppose there is a pseudorandom function generator against $\mathfrak{B}$. There is a constant $c > 0$ such that if $\mathsf{MCSP}[s]$ is in $\mathfrak{C}$ on average with $\mathfrak{B}$-bounded success ε, then $\mathsf{MCSP}[s + n^c, s - n^c]$ is in $\mathfrak{C}$.*

Proof. By assumption, there is a pseudorandom function generator $\{F_n\}$ against $\mathfrak{B}$; without loss of generality, the sequence of functions $\{G_n\}$ corresponding to this generator is computable in size $< n^c$ for some constant c. Let A be the $\mathfrak{C}$-algorithm solving $\mathsf{MCSP}[s]$ on average, and let A' be the $\mathfrak{B}$-algorithm bounding the error set of A. As in the proof of Theorem 23, there is

a pseudorandom reduction from $\mathsf{MCSP}[s+n^c, s-n^c]$ to $\mathsf{MCSP}[s]$ given by $g(y,r) = y \oplus F_n(r)$ and $h(y,r,b) = b$, where $|y| = 2^n$. The key idea is that because $\{F_n\}$ is pseudorandom against $\mathfrak{B}$, A' cannot distinguish the output of the pseudorandom reduction from a purely random string of the same length, and this means that with noticeable probability, the output of the reduction must fall outside the error set. More precisely, let S_N be the set of N-bit inputs on which A' outputs 0, i.e., S_N does not intersect the error set. The density of S_N is at least ε, and this means that the probability that the output of the pseudorandom reduction is in S_N is at least $\varepsilon - 1/N \geq 1/N$ by assumption on ε, for if not, $D(z) = A'(y \oplus z)$ would $1/N$-distinguish U_N from the distribution given by the pseudorandom function generator. By running the reduction $O(N)$ times independently, and each time simulating A on the output and outputting the answer if it is not '?', we get a $\mathfrak{C}$-algorithm for MCSP, using the assumed closure properties of $\mathfrak{C}$. ◀

▶ **Corollary 25.** *Suppose exponentially hard one-way functions exist. Let $s : \mathbb{N} \to \mathbb{N}$ be a size function such that $s(n) = n^{\omega(1)}$. There is a constant c such that if $\mathsf{MCSP}[s]$ is feasibly on average in SZK, then $\mathsf{MCSP}[s+n^c, s-n^c]$ is in SZK.*

Corollary 25 follows from Theorem 24 by using Theorem 21 and the fact that SZK is known to be closed under disjunctive truth-table reductions.

4 De Morgan Formula Lower Bounds for MCSP

In this section, we will prove an unconditional formula lower bound for computing MCSP.

▶ **Theorem 26.** $\mathsf{MCSP}[s]$ *requires a de Morgan formula of size $n^{2-O(1/\sqrt{\log n})}$ for $s(n) = n^{1/2\sqrt{\log n}}$.*

Throughout this section, Γ denotes the shrinkage exponent (i.e. $\Gamma = 2$ [17]). For a Boolean function f, $\mathrm{L}(f)$ denotes the minimum size of a de Morgan formula that computes f. We say that a random restriction $\rho\colon [n] \to \{0,1,*\}$ is p-regular if $\Pr[\rho(x_i) = *] = p$ for any $i \in [n]$.

4.1 A Review of Impagliazzo, Meka and Zuckerman [20]

The proof is based on the results by Impagliazzo, Meka and Zuckerman [20], which show that a pseudorandom restriction is enough to shrink de Morgan formulas:

▶ **Lemma 27** (Impagliazzo, Meka and Zuckerman [20])**.** *Let $f\colon \{0,1\}^n \to \{0,1\}$ and $p^\Gamma \mathrm{L}(f) \geq 1$. Let $\mathcal{R}_{p,l}$ be a distribution of p-regular l-wise random restrictions. Then, $\mathbb{E}_{\rho \sim \mathcal{R}_{p,l}}[\mathrm{L}(f)] \leq O(p^\Gamma \mathrm{L}(f))$ for $l := p^{-\Gamma}$.*

Proof Sketch (based on [25]). The idea is to decompose the formula of size $\mathrm{L}(f)$ into the small subformulas of size at most $l := p^{-\Gamma}$, which enables us to argue that each subformula shrinks under l-wise random restriction. Specifically, by using the fact that a tree of leaf size s can be decomposed into two trees each of which is of size between $s/3$ and $2s/3$ (as in the proof of Spira's theorem [33]), we can decompose a de Morgan formula computing f into subformulas $g_1, \ldots g_m$ such that $l/6 \leq \mathrm{L}(g_i) \leq l$ for each $i \in [m]$. Note that $m \leq 6\mathrm{L}(f)/l$. The variables of these subformulas consist of, in addition to the original variables of f, special variables[1] which refer to the other subtrees. Thus, we have $\mathrm{L}(f|_\rho) \leq \sum_{i=1}^m \mathrm{L}(g_i|_{\rho'})$ for any

[1] In this proof, we do not count the number of special variables in the size $\mathrm{L}(g_i)$ of subformulas.

restriction ρ, where ρ' denotes the restriction such that $\rho'(x_i) = *$ if x_i is a special variable and $\rho'(x_i) = \rho(x_i)$ otherwise. Furthermore, by some appropriate conversion, we may assume that each g_i has at most two special variables.

From now on the goal is, instead of upper bounding $\mathbb{E}_{\rho\sim\mathcal{R}_{p,l}}[\mathrm{L}(f|_\rho)]$, to bound $\mathbb{E}_{\rho\sim\mathcal{R}_{p,l}}[\mathrm{L}(g_i|_{\rho'})]$. By using a formula for computing the addressing function, we have $\mathrm{L}(g_i|_{\rho'}) \leq \sum_{\sigma\in\{0,1\}^S} (\mathrm{L}(g_i|_{\sigma\rho'}) + |S|)$, where S denotes the set of special variables in g_i and σ denotes an arbitrary assignment to special variables. Once the special variables are removed from g_i by applying a restriction σ, it holds that

$$\mathop{\mathbb{E}}_{\rho\sim\mathcal{R}_{p,l}}[\mathrm{L}(g_i|_{\sigma\rho'})] = \mathop{\mathbb{E}}_{\rho\sim\mathcal{R}_{p,\infty}}[\mathrm{L}(g_i|_{\sigma\rho'})] \leq p^\Gamma \mathrm{L}(g_i),$$

where the first equality holds because $g_i|_\sigma$ depends on at most l variables and the second equality holds because of the definition of the shrinkage exponent. To summarize,

$$\begin{aligned}
\mathop{\mathbb{E}}_{\rho\sim\mathcal{R}_{p,l}}[\mathrm{L}(f|_\rho)] &\leq \sum_{i=1}^m \mathop{\mathbb{E}}_{\rho\sim\mathcal{R}_{p,l}}[\mathrm{L}(g_i|_{\rho'})] \\
&\leq \sum_{i=1}^m \sum_{\sigma\in\{0,1\}^S} \left(\mathop{\mathbb{E}}_{\rho}[\mathrm{L}(g_i|_{\sigma\rho'})] + |S|\right) \\
&\leq \sum_{i=1}^m 4\left(p^\Gamma \mathrm{L}(g_i) + 2\right) \\
&\leq m \cdot 4(p^\Gamma l + 2) \\
&\leq m \cdot 4(p^\Gamma l + 2p^\Gamma l) && \text{(since } l \geq p^{-\Gamma}\text{)} \\
&= 12p^\Gamma ml \\
&\leq 72p^\Gamma \mathrm{L}(f) && \text{(since } m \leq 6\mathrm{L}(f)/l\text{)}\,. \qquad \blacktriangleleft
\end{aligned}$$

In the standard situation, we set $p = n^{-\Omega(1)}$, and hence we require as large independence as $n^{\Omega(1)}$-wise in the previous lemma. However, we can significantly reduce the number of random bits needed to generate pseudorandom restrictions by composing $l = 2^{O(\sqrt{\log n})}$-wise independent random restriction $r = O(\sqrt{\log n})$ times:

▶ **Theorem 28** (Impagliazzo, Meka and Zuckerman [20]). *Let $f\colon \{0,1\}^n \to \{0,1\}$ and $p^\Gamma \mathrm{L}(f) \geq 1$. Let $q = p^{1/r}$ for some nonnegative integer $r \geq 1$. Let $\mathcal{R}^r_{p,l}$ be a distribution of the composition of r independent q-regular l-wise random restrictions. (Hence, the composed random restriction is p-regular.) Then, $\mathbb{E}_{\rho\sim\mathcal{R}^r_{p,l}}[\mathrm{L}(f)] \leq c^r p^\Gamma \mathrm{L}(f)$ for $l := q^{-\Gamma}$ and for some constant c.*

Proof. By induction on $r \geq 1$. Let $c = 72$ be the universal constant in Lemma 27. The base case is exactly the same with Lemma 27. Now let us assume $r > 1$. Fix a composition $\rho_0 \in \mathrm{supp}(\mathcal{R}^{r-1}_{p,l})$ of $r-1$ restrictions. We pick a l-wise independent random restriction $\rho_1 \sim \mathcal{R}_{p,l}$. By applying Lemma 27 for $f|_{\rho_0}$, we obtain

$$\mathop{\mathbb{E}}_{\rho_1\sim\mathcal{R}_{p,l}}[\mathrm{L}(f|_{\rho_0\rho_1})] \leq cq^\Gamma \mathrm{L}(f|_{\rho_0}).$$

By averaging this inequality under distribution $\rho_0 \sim \mathcal{R}^{r-1}_{p,l}$, it holds that

$$\begin{aligned}
\mathop{\mathbb{E}}_{\rho\sim\mathcal{R}^r_{p,l}}[\mathrm{L}(f)] &\leq cq^\Gamma \mathop{\mathbb{E}}_{\rho_0\sim\mathcal{R}^{r-1}_{p,l}}[\mathrm{L}(f|_{\rho_0})] \\
&\leq cq^\Gamma \cdot c^{r-1} q^{\Gamma(r-1)} \mathrm{L}(f) && \text{(by the induction hypothesis)} \\
&= c^r p^\Gamma \mathrm{L}(f). \qquad \blacktriangleleft
\end{aligned}$$

4.2 Proof of Theorem 26

Now we are ready to prove the unconditional formula lower bound for MCSP. First, note that a q-regular l-wise independent random restriction can be sampled by using random $O(l \log n \log \frac{1}{q})$ bits, and that each coordinate of a random restriction can be computed in time a polynomial in the number of random bits (see [11]). Hence, the output of a composition of r q-regular l-wise independent random restrictions has circuit complexity at most $s := \mathrm{poly}(r, l, \log n, \log \frac{1}{q})$ when regarded as a truth table. The circuit complexity s is significantly smaller than the expected number pn of the unrestricted inputs under p-regular random restrictions, for some appropriate parameters. Specifically, let $p := 2^{\sqrt{\log n}}/n$, $q := p^{1/r}$, $l := q^{-\Gamma}$ and $r := C\sqrt{\log n}$ for some large constant C so that $s = \mathrm{poly}(r, p^{-\Gamma/r}, \log n, \log \frac{1}{p}) \leq 2^{\frac{1}{2}\sqrt{\log n}} \ll pn$.

By Theorem 28, we have $\mathbb{E}_{\rho \sim \mathcal{R}^r_{p,l}}[\mathrm{L}(f|_\rho)] \leq c^r p^\Gamma \mathrm{L}(f)$. Hence, the goal is to obtain a lower bound on $\mathbb{E}_{\rho \sim \mathcal{R}^r_{p,l}}[\mathrm{L}(f|_\rho)]$. We claim that a pseudorandom restriction does not shrink the formula for computing MCSP:

▶ **Lemma 29.** *Let $\rho\colon [n] \to \{0, 1, *\}$ be a restriction such that ρ can be computed by a circuit of size s, and let $V := \rho^{-1}(*)$. Then, for $f = \mathsf{MCSP}[s]$, we have $\mathrm{L}(f|_\rho) \geq |V| - O(s \log s)$.*

Proof. Let $V_0 \subset V$ be the set of variables on which $f|_\rho$ does not depend. It suffices to claim that $|V_0| = O(s \log s)$ because $\mathrm{L}(f|_\rho) \geq |V| - |V_0|$.

Indeed, let $\sigma\colon V \to \{0, 1\}$ denote an assignment for variables in V. For $\sigma \equiv 0$, the circuit size of the truth table $\rho \circ \sigma \in \{0, 1\}^n$ is at most s. Hence, $\rho \circ \sigma$ is an YES instance of $\mathsf{MCSP}[s]$. Since $f|_\rho$ does not depend on V_0, any assignment σ such that $V \setminus V_0 \subset \sigma^{-1}(0)$ is also an YES instance of $\mathsf{MCSP}[s]$. The number of such assignments is $2^{|V_0|}$, whereas the number of circuits of size at most s is $s^{O(s)}$. Therefore, we have $2^{|V_0|} \leq 2^{O(s \log s)}$. ◀

We can easily show that $|\rho^{-1}(*)| \geq pn/2$ with probability at least $\frac{1}{2}$ by using pairwise independence of ρ and Chebyshev's inequality. Therefore,

$$\begin{aligned}\mathop{\mathbb{E}}_{\rho \sim \mathcal{R}^r_{p,l}}[\mathrm{L}(f|_\rho)] &\geq \Pr_\rho\left[|\rho^{-1}(*)| \geq \frac{pn}{2}\right] \cdot \mathop{\mathbb{E}}_{\rho \sim \mathcal{R}^r_{p,l}}\left[\mathrm{L}(f|_\rho) \;\middle|\; |\rho^{-1}(*)| \geq \frac{pn}{2}\right] \\ &\geq \frac{1}{2} \cdot \left(\frac{pn}{2} - O(s \log s)\right) \geq \frac{pn}{8},\end{aligned}$$

where the last inequality holds since $O(s \log s) \ll pn$. Thus, $\frac{pn}{8} \leq \mathbb{E}_\rho[\mathrm{L}(f|_\rho)] \leq c^r p^\Gamma \mathrm{L}(f)$ and hence $\mathrm{L}(f) \geq np^{-\Gamma+1}/8c^{-r} = n^2 \cdot 2^{-O(\sqrt{\log n})}$.

5 Average-case $\mathsf{AC}^0[p]$ Lower Bound of MKTP

In this section, we show an unconditional average-case $\mathsf{AC}^0[p]$ circuit lower bound of MKTP. Our result improves a previous result [8] showing a worst-case $\mathsf{AC}^0[p]$ circuit lower bound of MKTP. The whole section is devoted to proving the following result:

▶ **Theorem 30.** *There exists some function $s(n)$ such that $\mathsf{MKTP}[s]$ is not in $\mathsf{AC}^0[p]$ on average with error ϵ, for any prime p and any constant $\epsilon \in (0, 1)$.*

Our proof is based on the techniques of Fefferman, Shaltiel, Umans and Viola [14] They gave a pseudorandom generator against $\mathsf{AC}^0[p]$ that is implicitly computable (i.e. each output bit of the pseudorandom generator is easy to compute, or in other words, the KT-complexity is small). We first focus on the case when $p \neq 2$. In this case, we use the following pseudorandom generator G based on PARITY.

▶ **Definition 31** ([14]). Define $G\colon (\{0,1\}^n)^k \to \{0,1\}^{nk+k}$ as

$$G(x_1,\dots,x_k) := x_1 \cdots x_k \cdot \mathsf{PARITY}(x_1) \cdots \mathsf{PARITY}(x_k)$$

for $(x_1,\dots,x_k) \in (\{0,1\}^n)^k$.

▶ **Lemma 32** (implicit in [14]). *If there is an oracle A that distinguishes G from the uniform distribution with advantage a constant $\epsilon > 0$, then there is an AC^0 circuit C with A-oracle gates such that $C^A(x) = \mathsf{PARITY}(x)$ for any $x \in \{0,1\}^n$.*

Proof Sketch. For completeness, we include a brief proof sketch. They showed that, by using *resamplability* of PARITY, there is an NC^0 circuit C_0 with one A-oracle gate such that $\Pr_{x\sim\mathcal{U}_n}[C_0^A(x) = \mathsf{PARITY}(x)] \geq \frac{1+\epsilon}{2}$ ([14, Lemma 4.5]). By using resamplability again for t independent choices of randomness (for some appropriately chosen t), we obtain circuits $C_1^A,\dots,C_t^A$ each of which approximates PARITY. Now taking the majority vote of these circuits, we can compute PARITY on all inputs. Here, the majority can be implemented by using Approximate Majority [1] in AC^0, because the advantage of approximating PARITY is at least a constant ϵ. As a result, we obtain an AC^0 circuit with A-oracle gates that computes PARITY on all inputs ([14, Proposition 4.21]). ◀

Therefore, it is sufficient to claim that an average-case easiness of $\mathsf{MKTP}[s]$ implies that the pseudorandom generator G can be broken. We first claim that the KT-complexity of any output of the pseudorandom generator G in Lemma 32 is small.

▶ **Claim 33.** $\mathrm{KT}(G(x_1,\dots,x_k)) \leq nk + n \cdot \mathsf{polylog}(n)$ *for any seed* $(x_1,\dots,x_k) \in (\{0,1\}^n)^k$.

Proof. We use a description $d := (x_1,\dots,x_k)$. Given an index $i \in \{1,\dots nk+k\}$ of $G(x_1,\dots,x_k)$, if $i \leq nk$ then output the ith bit of the description d; if $i > nk$ then compute and output $\mathsf{PARITY}(x_{i-nk})$, which takes $O(n)$ steps. A universal machine simulates this computation in time $n \cdot \mathsf{polylog}(n)$. ◀

Therefore, for $k := n^3$, it holds that $\mathrm{KT}(G(x_1,\dots,x_k)) \leq nk + o(k)$ (and thus an MKTP oracle distinguishes G from the uniform distribution).

Now let us assume, towards a contradiction, that there is an $\mathsf{AC}^0[p]$ circuit A_0 that computes $\mathsf{MKTP}[s]$ all but an ϵ fraction of inputs with zero-sided error. We define another circuit A as $A(x) := 1$ if $A_0(x) = 1$ or ?; otherwise $A(x) := 0$. Note that A does not err on yes instances of $\mathsf{MKTP}[s]$. We claim that that A breaks G.

▶ **Claim 34.** *Let $s(n) := n - \sqrt{n}$. The following holds.*

1. $\Pr[A(G(\mathcal{U}_n,\dots,\mathcal{U}_n)) = 1] = 1$.
2. $\Pr[A(\mathcal{U}_{nk+k}) = 1] \leq \epsilon + o(1)$.

Proof.

1. By Claim 33, for any $y = G(x_1,\dots,x_k) \in \{0,1\}^{nk+k}$, we have $\mathrm{KT}(y) = nk + \widetilde{O}(n) \ll n^4 + n^3 - \sqrt{n^4+n^3} = s(nk+k)$; hence, y is an yes instance of $\mathsf{MKTP}[s]$ and $A(y) = 1$.
2. The point is that, under the uniform distribution, there are few yes instances in $\mathsf{MKTP}[s]$. Hence, the algorithm A that solves $\mathsf{MKTP}[s]$ on a $1-\epsilon$ fraction of instances must have a substantial fractions of no instances on which A succeeds. Formally,

$$\begin{aligned}
\Pr_{x\sim\mathcal{U}_{nk+k}}[A(x) = 0] &= \Pr_x[A_0(x) = 0] \\
&= \Pr_x[A_0(x) \neq ? \ \wedge \ x \notin \mathsf{MKTP}[s]] \\
&\geq \Pr_x[A_0(x) \neq ?] - \Pr_x[x \in \mathsf{MKTP}[s]] \\
&\geq 1 - \epsilon - 2^{-\sqrt{nk+k}}.
\end{aligned}$$

◀

In particular, A distinguishes the output of G from the uniform distribution with advantage $1-\epsilon-o(1) \geq \frac{1-\epsilon}{2}$. Now we apply Lemma 32 to obtain an AC^0 circuit C^A with A-oracle gates that solves PARITY. Since $A \in \mathsf{AC}^0[p]$, it shows that $\mathsf{PARITY} \in \mathsf{AC}^0[p]$, which contradicts the lower bounds of Razborov-Smolensky [29, 32] for odd prime p.

When $p = 2$, we use a pseudorandom generator G_{CMD} based on a problem called CMD (connectivity matrix determinant), which was introduced by Ishai and Kushilevitz [21, 22]. For the exact definition of G_{CMD}, the reader is referred to [14]. Here we only need the following property, which easily follows from the fact that CMD is computable in polynomial time.

▶ **Fact 35** (Revised Claim 33). $\mathrm{KT}(G_{\mathsf{CMD}}(x_1, \ldots, x_k)) \leq nk + n^{O(1)}$ *for any seed* $(x_1, \ldots, x_k) \in (\{0,1\}^n)^k$.

▶ **Lemma 36** ([14]). *If there is an oracle A that distinguishes G_{CMD} from the uniform distribution with advantage a constant $\epsilon > 0$, then there is an $\mathsf{AC}^0[2]$ circuit C with A-oracle gates such that $C^A(x) = \mathsf{MAJORITY}(x)$ for any $x \in \{0,1\}^n$.*

Proof Sketch. The problem CMD is resamplable in $\mathsf{AC}^0[2]$ ([14]), and hence as in Lemma 32, CMD can be solved by an $\mathsf{AC}^0[2]$ circuit with A-oracle gates. Since CMD is $\oplus\mathsf{L}$-complete under NC^0 reductions ([22]), $\mathsf{MAJORITY}$ can be also solved by an $\mathsf{AC}^0[2]$ circuit with A-oracle gates. ◀

Combining Fact 35 and Lemma 36, we obtain an $\mathsf{AC}^0[2]$ circuit that solves $\mathsf{MAJORITY}$, which contradicts the lower bound of [29, 32] for the majority function. This completes the proof of Theorem 30.

6 MKTP and Average-case Hardness Conjectures

In this section, we show hardness of MKTP and MCSP under popular hypotheses on average-case hardness of various problems.

6.1 Random 3SAT Hardness of MKTP

Let us consider the distribution of a random 3CNF formula on n variables such that the formula is the conjunction of $m = \Delta n$ clauses sampled from all the possible $2^3\binom{n}{3}$ 3-literal clauses independently and uniformly at random. Given such a formula, Feige's hypothesis states that there is no polynomial-time algorithm that (1) accepts every formula for which all but ϵm clauses are satisfiable (henceforth, call such a formula ϵ-almost satisfiable), and (2) rejects most formulas (i.e. with probability $\frac{1}{2}$ over the choice of a random 3CNF formula).

▶ **Hypothesis 37** (Feige [15]). *For every fixed $\epsilon > 0$ and sufficiently large constant Δ (which are independent of n), there is no polynomial time algorithm that accepts every ϵ-almost satisfiable formula, and rejects most formulas.*

Note that there is a variant of the hypothesis stating that there is no polynomial time algorithm that accepts every *satisfiable* formula and rejects most formulas. This variant is stronger than Hypothesis 37 and may be sensitive to minor model changes (see [15] for more details). Here we refute the weaker hypothesis under MKTP oracle, and hence our result is stronger.

▶ **Theorem 38.** MKTP *is random 3SAT-hard in the sense of* [15]. *That is, there is a polynomial-time algorithm with oracle access to* MKTP *that refutes Hypothesis 37.*

Proof. We construct a many-one reduction from random 3SAT to MKTP. The reduction is simple: given a formula φ, map it to (φ, θ) for some threshold θ chosen later. The idea is that any ϵ-almost satisfiable formula is atypical, and hence it can be described efficiently given an almost satisfying assignment (i.e. the KT-complexity of any ϵ-almost satisfiable formula is small). More specifically, given an assignment x that satisfies all but ϵm clauses of the formula φ, each clause of φ that is satisfied by x has $(2^3-1)\binom{n}{3}$ possibilities; hence, each clause of φ (except for ϵm clauses) is of description length at most $\log 7\binom{n}{3}$. On the other hand, random 3SAT instance are chosen from the space of cardinality $[2^3\binom{n}{3}]^m$, and thus it has KT-complexity roughly $m \log 8\binom{n}{3}$ with high probability. Hence, the MKTP oracle enables us to distinguish ϵ-almost satisfiable formulas from random formulas, by exploiting the difference $m \log 8\binom{n}{3} - m \log 7\binom{n}{3}$ in KT-complexity. Details follow.

Define $\theta := m \log 8\binom{n}{3} - m/20$. We first claim that a random 3SAT formula has KT-complexity at least θ with high probability. Indeed, the number of strings with KT-complexity less than θ is at most 2^θ by simple counting. Thus, since a random 3SAT formula φ is chosen uniformly at random out of the space of cardinarity $[2^3\binom{n}{3}]^m = 2^{\theta + m/20}$, the probability that $\mathrm{KT}(\varphi) < \theta$ is at most $2^{-m/20}$.

The rest of the proof is devoted to proving ϵ-almost satisfiable formula is of low KT-complexity:

▶ **Claim 39.** *For sufficiently small $\epsilon > 0$ and any ϵ-almost satisfiable formula φ, $\mathrm{KT}(\varphi) < \theta$.*

In order to claim that the KT-complexity of φ is small, we need to implement an efficient procedure that, given an index, outputs the clause of φ specified by the index, with random access to a description of φ. We will describe φ by using an ϵ-almost satisfying assignment $x \in \{0,1\}^n$, a subset $S \in \binom{[m]}{\leq \epsilon m}$ of clauses not satisfied by x, $(1-\epsilon)m \log 7\binom{n}{3}$ bits to describe clauses satisfied by x, and $\epsilon m \log \binom{n}{3}$ bits to describe clauses not satisfied by x.

In order to describe each clause of φ efficiently (i.e. in time $\mathsf{polylog}(m)$), there are two issues for which we need ideas from succinct data structures. One is an efficient representation of S. Information theoretically, S can be described in $\log \binom{m}{\epsilon m} \leq \epsilon m \log(em/\epsilon m) = m\epsilon \log(e/\epsilon) < m/100$ bits for sufficiently small $\epsilon > 0$. However, a naive representation of S may not enable us to answer a query $i \stackrel{?}{\in} S$ efficiently; thus, we need the following result.

▶ **Lemma 40** (Brodnik and Munro [13])**.** *There exists a string d_S of length $\log \binom{m}{\epsilon m} + o(\log \binom{m}{\epsilon m})$ and an algorithm that, given random access to d_S and index i, answers a query $i \stackrel{?}{\in} S$ in time $\mathsf{polylog}(m)$.*

The other issue is the use of the ceiling function (c.f. [28, 3]). For each clause satisfied by x, we need $\lceil \log 7\binom{n}{3} \rceil$ bits (if we represent each clause separately), which is not necessarily smaller than $\log 8\binom{n}{3}$ bits. We thus group consecutive $b := 11$ clauses of φ into one block so that each block encodes b clauses by using at most $\lceil b \log 7\binom{n}{3} \rceil$ bits. Since $(\frac{7}{8})^b \leq \frac{1}{4}$, we have $\lceil b \log 7\binom{n}{3} \rceil \leq b \log 8\binom{n}{3} - 1$; thus, we can dispense with 1 bit for each block.

Hence, the KT-complexity of φ is

$$\begin{aligned}
\mathrm{KT}(\varphi) &\leq n + \log \binom{m}{\epsilon m} + o\left(\log \binom{m}{\epsilon m}\right) + \left\lceil \frac{m}{b} \right\rceil \cdot \left\lceil b \log 7\binom{n}{3} \right\rceil + \mathsf{polylog}(m) \\
&\leq \frac{m}{\Delta} + \frac{m}{100} + m \log 8\binom{n}{3} - \frac{m}{b} + o(m) \\
&\leq m \log 8\binom{n}{3} - \frac{m}{20} = \theta
\end{aligned}$$

for sufficiently large Δ and m. ◀

Recently Ryan O'Donnell (personal communication) conjectured a co-nondeterministic version of Feige's hypothesis, i.e., that Hypothesis 37 holds even with respect to co-nondeterministic polynomial-time algorithms. It follows from the proof of Theorem 38 that MKTP is not in coNP under O'Donnell's conjecture. This is the first evidence of any kind that MCSP or MKTP is not in coNP. There has been some speculation about whether MCSP $\in$ SZK, for example this is posed as an open problem in Allender's recent survey [4]. Under a standard derandomization hypothesis, SZK $\subseteq$ NP $\cap$ coNP, hence if MKTP $\in$ SZK, either this hypothesis fails or O'Donnell's conjecture fails.

It should be noted that our proof does not seem to carry over to the case of MCSP. The gap between the KT-complexity of almost satisfiable formulas and random formulas is smaller than $m = o(|\varphi|)$, and it is not clear how to construct a small circuit which simulates the random access machine with additive overhead smaller than m. We leave as an open question to extend Theorem 38 to the case of MCSP.

6.2 Hardness of MCSP under Alekhnovich's hypothesis

While we were not able to prove that MCSP is random 3SAT-hard, we can refute a strong hypothesis about average-case complexity proposed by Alekhnovich [2] under MCSP oracle. He considered a problem of solving linear equations under a certain noise e. Let A be an $m \times n$ matrix over GF(2). Let $D_k(A)$ be the distribution of a random vector $Av + e$, where v is a uniform sample from $\mathrm{GF}(2)^n$ and $e \in \mathrm{GF}(2)^n$ is a uniform sample from the vectors of Hamming weight k (i.e. the number of ones in e is k). Alekhnovich conjectured that there is a matrix such that it is infeasible to distinguish $D_k(A)$ from $D_{k+1}(A)$ efficiently.

▶ **Hypothesis 41** (Alekhnovich [2, Conjecture 4.5])**.** *For every $m(n) = \Theta(n)$, there exists a family of $m(n) \times n$ matrices $\{A_n\}_{n\in\mathbb{N}}$ such that, for every function $k(n)$ which satisfies $n^\epsilon < k(n) < n^{1-\epsilon}$ for some constant $\epsilon > 0$, for every efficient algorithm M, the success probability*

$$|\Pr[M(D_k(A_n)) = 1] - \Pr[M(D_{k+1}(A_n)) = 1]|$$

is negligible.

▶ **Theorem 42.** *There is a polynomial-time algorithm with oracle access to* MCSP *that refutes Hypothesis 41.*

Proof Sketch. Alekhnovich showed that Hypothesis 41 implies the existence of a cryptographic pseudorandom generator ([2, Lemma 4.14]). Now we can construct a pseudorandom function generator as in [16], based on Hypothesis 41. On the other hand, an MCSP oracle can distinguish the output distribution of the pseudorandom function generator from the uniform distribution (see, e.g., [5]). Hence, there is an efficient algorithm that refutes Hypothesis 41 with oracle access to MCSP. ◀

6.3 Planted Clique Hardness of MKTP

Now we move on to planted clique conjectures [23, 26].

Let $G(n, \frac{1}{2})$ denote the distribution of an n-vertex graph whose edges are placed with probability $\frac{1}{2}$ independently (i.e. an Erdős-Rényi random graph). Let $G(n, \frac{1}{2}, k)$ be the distribution of a random graph such that a graph is chosen from $G(n, \frac{1}{2})$ and then a clique of size k is randomly placed in the graph. The decision version of planted clique conjectures states that there is no polynomial time algorithm that distinguishes $G(n, \frac{1}{2}, k)$ from $G(n, \frac{1}{2})$.

On average, there is a clique of size $2\log n$ on $G(n, \frac{1}{2})$, and thus there is a quasipolynomial-time algorithm for solving the planted clique problem by a brute force search. We show that there is a polynomial-time algorithm with oracle access to MKTP that solves the planted clique problem.

▶ **Theorem 43.** *For any $k \geq \mathsf{polylog}(n)$, there is a polynomial-time algorithm with oracle access to* MKTP *that accepts every graph chosen from $G(n, \frac{1}{2}, k)$, and rejects most random graphs chosen from $G(n, \frac{1}{2})$.*

Proof. The idea is the same with the proof of random 3SAT-hardness. As a many-one reduction from the planted clique problem to MKTP, given a random graph G, we map G to (G, θ) for a certain parameter θ.

We first claim that the KT-complexity of most random graphs G chosen from $G(n, \frac{1}{2})$ is large. Indeed, the graph is chosen uniformly at random from the space of cardinarity $2^{\binom{n}{2}}$; thus, the probability that $\mathrm{KT}(G)$ is less than $\binom{n}{2} - k$ is at most $2^{-k} = 1/n^{\omega(1)}$, which is negligible. Define $\theta := \binom{n}{2} - k$.

Next, we claim that the KT-complexity of any graph G with k-clique is less than θ. For this purpose, we present an efficient algorithm that, on input a pair (v, w) of vertices and random access to a description, outputs whether G has an edge between v and w. Let S be a k-clique of G. The description for G consists of the clique S (which is encoded in $k \log n$ bits as the sorted list of vertices in S), and the adjacency matrix of G except for edges connecting vertices in S (which can be encoded in $\binom{n}{2} - \binom{k}{2}$ bits). The algorithm for describing G is as follows: Given the description and a pair (v, w), we first check whether $v \in S$ and $w \in S$ by a binary search. If v and w are in S, then we claim that there is an edge (since S is a clique). Otherwise, we compute an index of the description of an adjacency matrix to which (v, w) corresponds (which can be done in $\mathsf{polylog}(n)$ time), and then output the corresponding bit of the description.

The length of the description is roughly $k \log n + \binom{n}{2} - \binom{k}{2} \ll \theta$, and the time it takes to describe each bit of G is at most $\mathsf{polylog}(n)$. Hence, $\mathrm{KT}(G) < \theta$ for any G with a k-clique. ◀

References

1. Miklós Ajtai and Michael Ben-Or. A theorem on probabilistic constant depth computations. In *Proceedings of the 16th Annual ACM Symposium on Theory of Computing (STOC)*, pages 471–474, 1984. doi:10.1145/800057.808715.
2. Michael Alekhnovich. More on average case vs approximation complexity. *Computational Complexity*, 20(4):755–786, 2011. doi:10.1007/s00037-011-0029-x.
3. E. Allender, Joshua Grochow, and Cristopher Moore. Graph isomorphism and circuit size. Technical Report TR15-162, Electronic Colloquium on Computational Complexity, 2015.
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. doi:10.1007/978-3-319-50062-1_6.
5. Eric Allender, Harry Buhrman, Michal Koucký, Dieter van Melkebeek, and Detlef Ronneburger. Power from random strings. *SIAM J. Comput.*, 35(6):1467–1493, 2006. doi:10.1137/050628994.
6. Eric Allender and Bireswar Das. Zero knowledge and circuit minimization. In *Symposium on Mathematical Foundations of Computer Science* (MFCS), pages 25–32, 2014. doi:10.1007/978-3-662-44465-8_3.
7. Eric Allender, Lisa Hellerstein, Paul McCabe, Toniann Pitassi, and Michael E. Saks. Minimizing disjunctive normal form formulas and AC^0 circuits given a truth table. *SIAM J. Comput.*, 38(1):63–84, 2008. doi:10.1137/060664537.

8 Eric Allender and Shuichi Hirahara. New insights on the (non)-hardness of circuit minimization and related problems. *Electronic Colloquium on Computational Complexity (ECCC)*, 24:73, 2017. URL: https://eccc.weizmann.ac.il/report/2017/073.

9 Eric Allender, Dhiraj Holden, and Valentine Kabanets. The minimum oracle circuit size problem. In *Symposium on Theoretical Aspects of Computer Science* (STACS), volume 30 of *LIPIcs*, pages 21–33, 2015. doi:10.4230/LIPIcs.STACS.2015.21.

10 Eric Allender, Michael Koucky, Detlef Ronneburger, and Sambuddha Roy. The pervasive reach of resource-bounded kolmogorov complexity in computational complexity theory. *Journal of Computer and System Sciences*, 77:14–40, 2010.

11 Noga Alon, László Babai, and Alon Itai. A fast and simple randomized parallel algorithm for the maximal independent set problem. *J. Algorithms*, 7(4):567–583, 1986. doi:10.1016/0196-6774(86)90019-2.

12 Andrej Bogdanov and Luca Trevisan. On worst-case to average-case reductions for NP problems. *SIAM Journal on Computing*, 36(4):1119–1159, 2006.

13 Andrej Brodnik and J. Ian Munro. Membership in constant time and almost-minimum space. *SIAM J. Comput.*, 28(5):1627–1640, 1999. doi:10.1137/S0097539795294165.

14 Bill Fefferman, Ronen Shaltiel, Christopher Umans, and Emanuele Viola. On beating the hybrid argument. *Theory of Computing*, 9:809–843, 2013. doi:10.4086/toc.2013.v009a026.

15 Uriel Feige. Relations between average case complexity and approximation complexity. In *Proceedings on 34th Annual ACM Symposium on Theory of Computing (STOC)*, pages 534–543, 2002. doi:10.1145/509907.509985.

16 Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986. doi:10.1145/6490.6503.

17 Johan Håstad. The shrinkage exponent of de morgan formulas is 2. *SIAM J. Comput.*, 27(1):48–64, 1998. doi:10.1137/S0097539794261556.

18 Shuichi Hirahara and Osamu Watanabe. Limits of minimum circuit size problem as oracle. In *Conference on Computational Complexity* (CCC), volume 50 of *LIPIcs*, pages 18:1–18:20, 2016. doi:10.4230/LIPIcs.CCC.2016.18.

19 John M. Hitchcock and Aduri Pavan. On the NP-completeness of the minimum circuit size problem. In *Conference on Foundation of Software Technology and Theoretical Computer Science* (FSTTCS), volume 45 of *LIPIcs*, pages 236–245, 2015. doi:10.4230/LIPIcs.FSTTCS.2015.236.

20 Russell Impagliazzo, Raghu Meka, and David Zuckerman. Pseudorandomness from shrinkage. In *Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 111–119, 2012. doi:10.1109/FOCS.2012.78.

21 Yuval Ishai and Eyal Kushilevitz. Randomizing polynomials: A new representation with applications to round-efficient secure computation. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 294–304, 2000. doi:10.1109/SFCS.2000.892118.

22 Yuval Ishai and Eyal Kushilevitz. Perfect constant-round secure computation via perfect randomizing polynomials. In *Proceedings of Automata, Languages and Programming, 29th International Colloquium (ICALP)*, pages 244–256, 2002. doi:10.1007/3-540-45465-9_22.

23 Mark Jerrum. Large cliques elude the metropolis process. *Random Structures and Algorithms*, 3:347–359, 1992.

24 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.

25 Ilan Komargodski, Ran Raz, and Avishay Tal. Improved average-case lower bounds for De-Morgan formula size. In *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 588–597, 2013. doi:10.1109/FOCS.2013.69.

26 Ludek Kucera. Expected complexity of graph partitioning problems. *Discrete Applied Mathematics*, 57(2-3):193–212, 1995.

27 Cody D. Murray and Richard Ryan Williams. On the (non) NP-hardness of computing circuit complexity. In *Conference on Computational Complexity* (CCC), volume 33 of *LIPIcs*, pages 365–380, 2015. doi:10.4230/LIPIcs.CCC.2015.365.

28 Mihai Patrascu. Succincter. In *In Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 305–313, 2008. doi:10.1109/FOCS.2008.83.

29 Alexander Razborov. Lower bounds on the size of bounded depth circuits over a complete basis with logical addition. *Mathematical notes of the Academy of Sciences of the USSR*, 41(4):333–338, 1987. doi:10.1007/BF01137685.

30 Alexander A. Razborov and Steven Rudich. Natural proofs. *J. Comput. Syst. Sci.*, 55(1):24–35, 1997. doi:10.1006/jcss.1997.1494.

31 Steven Rudich. Super-bits, demi-bits, and NP/qpoly-natural proofs. In *Proceedings of Randomization and Approximation Techniques in Computer Science (RANDOM)*, pages 85–93, 1997. doi:10.1007/3-540-63248-4_8.

32 Roman Smolensky. Algebraic methods in the theory of lower bounds for boolean circuit complexity. In Alfred V. Aho, editor, *STOC*, pages 77–82. ACM, 1987. URL: http://dblp.uni-trier.de/db/conf/stoc/stoc87.html#Smolensky87.

33 Philip M Spira. On time-hardware complexity tradeoffs for boolean functions. In *Proceedings of the 4th Hawaii Symposium on System Sciences*, pages 525–527, 1971.

34 Boris Trakhtenbrot. A survey of russian approaches to perebor (brute-force search) algorithms. *IEEE Annals of the History of Computing*, 6(4):384–400, 1984.

35 Ryan Williams. Improving exhaustive search implies superpolynomial lower bounds. *SIAM J. Comput.*, 42(3):1218–1244, 2013. doi:10.1137/10080703X.

Easiness Amplification and Uniform Circuit Lower Bounds*

Cody D. Murray[1] and R. Ryan Williams[2]

1 **CSAIL and EECS, MIT, Cambridge, MA, USA**
`cdmurray@mit.edu`
2 **CSAIL and EECS, MIT, Cambridge, MA, USA**
`rrw@mit.edu`

Abstract

We present new consequences of the assumption that time-bounded algorithms can be "compressed" with non-uniform circuits. Our main contribution is an "easiness amplification" lemma for circuits. One instantiation of the lemma says: if $n^{1+\varepsilon}$-time, $\tilde{O}(n)$-space computations have $n^{1+o(1)}$ size (non-uniform) circuits for some $\varepsilon > 0$, then every problem solvable in *polynomial* time and $\tilde{O}(n)$ space has $n^{1+o(1)}$ size (non-uniform) circuits as well. This amplification has several consequences:

- **An easy problem without small LOGSPACE-uniform circuits.** For all $\varepsilon > 0$, we give a natural decision problem General Circuit n^{ε}-Composition that is solvable in $n^{1+\varepsilon}$ time, but we prove that polynomial-time and logarithmic-space preprocessing cannot produce $n^{1+o(1)}$-size circuits for the problem. This shows that there are problems solvable in $n^{1+\varepsilon}$ time which are not in LOGSPACE-uniform $n^{1+o(1)}$ size, the first result of its kind. We show that our lower bound is non-relativizing, by exhibiting an oracle relative to which the result is false.
- **Problems without low-depth LOGSPACE-uniform circuits.** For all $\varepsilon > 0$, $1 < d < 2$, and $e < d$ we give another natural circuit composition problem computable in $\tilde{O}(n^{1+\varepsilon})$ time, or in $O((\log n)^d)$ space (though not necessarily simultaneously) that we prove does not have SPACE$[(\log n)^e]$-uniform circuits of $\tilde{O}(n)$ size and $O((\log n)^e)$ depth. We also show SAT does not have circuits of $\tilde{O}(n)$ size and $\log^{2-o(1)} n$ depth that can be constructed in $\log^{2-o(1)} n$ space.
- **A strong circuit complexity amplification.** For every $\varepsilon > 0$, we give a natural problem Circuit n^{ε}-Composition and show that if it has $\tilde{O}(n)$-size circuits (uniform or not), then every problem solvable in $2^{O(n)}$ time and $2^{O(\sqrt{n \log n})}$ space (simultaneously) has $2^{O(\sqrt{n \log n})}$-size circuits (uniform or not). We also show the same consequence holds assuming SAT has $\tilde{O}(n)$-size circuits.

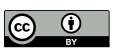
  As a corollary, if $n^{1.1}$ time computations (or $O(n)$ nondeterministic time computations) have $\tilde{O}(n)$-size circuits, then *all* problems in exponential time and subexponential space (such as quantified Boolean formulas) have *significantly* subexponential-size circuits. This is a new connection between the relative circuit complexities of easy and hard problems.

1998 ACM Subject Classification F.1.3 Relations Among Complexity Classes

Keywords and phrases uniform circuit complexity, time complexity, space complexity, non-relativizing, amplification

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.8

* Supported by NSF CCF-1552651 (CAREER). Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.



32nd Computational Complexity Conference (CCC 2017).
Editor: Ryan O'Donnell; Article No. 8; pp. 8:1–8:21

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

1 Introduction

Boolean circuit complexity and machine-based computation are nicely bridged by the notion of *uniform circuits*. These are circuit families where any particular circuit in the family can be efficiently generated on demand, by an algorithm consuming few resources. We use $\mathcal{C}$-uniform $\mathsf{SIZE}[s(n)]$ to (informally) denote the class of problems computable by a circuit family $\{C_n\}$ of $s(n)$-size such that the description of C_n is computable in $\mathcal{C}$ for all n. (The choice of circuit description/encoding can vary, and the results of our paper do not depend on these choices. See Section 2 for formal definitions.) Borrowing some terminology from data structures, one can say that uniform circuits naturally capture preprocessing/query tradeoffs in a static model: putting a problem in $\mathcal{C}$-uniform $\mathsf{SIZE}[s(n)]$ means that we can preprocess with $\mathcal{C}$ resources to build a size-$s(n)$ circuit, so that every subsequent n-bit instance of the problem can be computed (queried) with that circuit.

Most work on uniform circuit complexity over the last several decades has studied either the case where the class $\mathcal{C}$ is extremely weak, or is extremely strong. The extremely weak uniform models, such as $\mathsf{LOGTIME}$-*uniform circuits*, are the closest to machine-based computation: any particular wire or gate of a size-s circuit C_n in the family can be computed in only $O(\log s)$ time, proportional to the sizes of pointers to the wires/gates. The measures of "$\mathsf{LOGTIME}$-uniform circuit size" and "time" are known to coincide up to polylogarithmic factors [22], hence there are problems solvable in $n^{1.002}$ that are not in $\mathsf{LOGTIME}$-uniform $\mathsf{SIZE}[n^{1.001}]$, by the time hierarchy theorem [13, 33]. On the other end of the uniformity spectrum, *non-uniform* circuits require no computable bounds on generating the circuits, and far less is known: for example, no functions in huge classes like $\mathsf{TIME}[2^{O(n)}]^{SAT}$ are known to require even $4n$-size non-uniform circuits over the basis of all Boolean functions on two inputs, although some progress has recently been made on this front [9].

A few years ago, Santhanam and Williams [25] studied so-called "medium-uniform" circuits, where the complexity of generating a circuit is neither very weak nor very strong, e.g., $\mathsf{LOGSPACE}$, P, and P^{NP}-uniformity. Among other results, Santhanam and Williams [25] proved that for some k, there is a problem solvable in n^k time that does not have P-*uniform* linear-size circuits; that is, they proved $\mathsf{P} \not\subset \mathsf{P}$-uniform $\mathsf{SIZE}[O(n)]$. (In fact, they proved the stronger result that for every c, there is a k_c and a problem in $\mathsf{TIME}[n^{k_c}]$ that does not have P-uniform n^c-size circuits.) That is, they prove a *super-polynomial* time lower bound on preprocessing linear-size circuits for solving a problem in P. (It is believed there are problems in P without linear-size circuits *period*, no matter how much preprocessing is used, but this is an infamously difficult and famous open problem.) Their techniques led to similar results for $\mathsf{LOGSPACE}$-uniform branching programs, and lower bounds for NP versus $\mathsf{P}^{\mathsf{NP}}_{||}$-uniform linear size circuits.

There are two major drawbacks of the lower bound method of Santhanam and Williams. The first is its extreme non-constructivity: their method is a (very) indirect diagonalization argument. No particular problem in n^k time is known to exhibit the circuit size lower bound, and in fact the proof provides *no explicit bound on* k. That is, we cannot point to any problem in P satisfying the lower bound; we cannot even point to an upper bound on the time complexity of such a problem. This non-constructive phenomenon holds for all lower bounds in their work, creating a frustrating state of affairs. The second more serious drawback of their method is that it *relativizes*, which implies that there are hard barriers to what it can possibly prove. In particular, we cannot expect to prove results like $\mathsf{P} \not\subset \mathsf{SIZE}[O(n)]$ via such techniques.

1.1 Our Results

We introduce a non-relativizing method for exploiting the weakness of small-space computations, and for "amplifying" assumptions on the circuit complexity of easy problems. Using this method, we identify natural circuit composition problems which can easily be solved in low-polynomial time, yet we can prove non-trivial LOGSPACE-uniform circuit lower bounds for computing them. That is, *no* small-space algorithm can generate small circuits for solving the problems, despite their tractability. Our techniques give new insight into how to prove limitations on small-space computation. It is possible that similar ideas could potentially apply to non-uniform models of small-space computation, such as branching programs.

▶ **Definition 1.** In the $k(n)$-IO Circuit t-Composition problem, we are given a Boolean circuit C over AND/OR/NOT of size n with $k(n)$ inputs and $k(n)$ outputs, an input $x \in \{0,1\}^{k(n)}$, and integer $t \geq 1$. The task is to output

$$C^t(x) := \underbrace{(C \circ \cdots \circ C)}_{t}(x),$$

i.e., C composed for t times on the input x.

The $k(n)$-IO Circuit t-Composition problem can also be expressed as a decision problem, by including an index $i = 1, \ldots, k(n)$ as input, and outputting the ith bit of $C^t(x)$.

When $k(n) = n^{o(1)}$, we simply call the problem Circuit t-Composition.

Observe that $k(n)$-IO Circuit t-Composition can be easily solved in $\tilde{O}(n \cdot t)$ time and $\tilde{O}(n)$ space, by straightforward simulation of the given size-n circuit for t times. (As is standard, we let $\tilde{O}(t(n))$ denote $t(n) \cdot (\log t(n))^c$ for an unspecified constant $c > 0$.)

The circuit composition problem defined above is a sequential "chain" of circuit evaluations. A more general version of the composition problem, which we call General Circuit t-Composition (defined in the Preliminaries), permits connections between multiple inputs and outputs of a given circuit; Circuit t-Composition is a special case of it. General Circuit t-Composition is also solvable in $\tilde{O}(n \cdot t)$ time (see Section 2); it also requires $\tilde{\Omega}(n \cdot t)$ time to be solved (see Theorem 14). (We let $\tilde{\Omega}(s(n))$ denote $s(n)/(\log s(n))^c$ for a constant $c > 0$.)

LOGSPACE-Uniform Circuit Lower Bounds

First, we prove circuit lower bounds for generically composing n^ε copies of a circuit, which is an $\tilde{O}(n^{1+\varepsilon})$-time task:

▶ **Theorem 2.** *For all $\varepsilon \in (0,1)$, General Circuit n^ε-Composition does not have $n^{1+o(1)}$-size circuits constructible in logarithmic space.*

We stress that Theorem 2 does *not* relativize. In Appendix A, we exhibit (for every constant $k \geq 1$) an oracle A such that every language in $\mathsf{TIME}(n^k)^A$ has $O(n)$-size A-oracle circuits constructible in $\mathsf{LOGSPACE}^A$.

It is well known that every problem solvable in t time *does* have LOGTIME-uniform circuits of $O(t \log t)$ size [22], and that there are problems solvable in t time that do not have LOGTIME-uniform $O(t/\log^3 t)$-size circuits, by the time hierarchy theorem [13, 14]. Theorem 2 is a significantly stronger lower bound than what is provided by the time hierarchy: it shows that *arbitrary* logarithmic space preprocessing (running in, say, $n^{10^{10^{10}}}$ time) is not enough to reduce the resources needed to solve Circuit n^ε-Composition even slightly less

than $\tilde{O}(n^{1+\varepsilon})$. That is, LOGSPACE-uniform circuits cannot be made noticeably smaller than the time-bounded computations they may simulate, in general.

Theorem 2 is interesting not only because it does not relativize, but because there are few non-trivial lower bounds known against LOGSPACE, even as a uniformity condition. For random-access models, it is known that SAT is not solvable in (simultaneous) $n^{1.8}$ time and $n^{o(1)}$ space, but there are concrete limitations on known proof methods [11, 30, 6]. Slightly non-linear time lower bounds (for $n^{1-\Omega(1)}$ space) are known for some functions in P [1, 5]. Fortnow [10] proved (along with follow-up work by [3, 28]) that SAT does not have LOGSPACE-uniform $n^{1+o(1)}$-size $O(\log n)$-depth circuits. Santhanam and Williams [25] proved there is a language in LOGSPACE which does not have LOGSPACE-uniform branching programs of $O(n^k)$ size for every k, but (as mentioned earlier) the argument yields no explicit language and no explicit resource bounds on the language. (On the other hand, small branching programs seem to be a weaker model than small $O(\log n)$-depth circuits.)

Super-Logarithmic Space Lower Bounds

Turning to the more restricted model of polylog-depth circuits, we can obtain stronger lower bounds than prior work. Informally, we define the problem $d(n)$-Depth Circuit t-Composition analogously to Circuit t-Composition, except the inputs are restricted to circuits of $d(n)$ depth and $d(n)$ input/output bits. The $d(n)$-Depth Circuit t-Composition problem can be solved in about $n \cdot t$ time (like Circuit t-Composition) or in about $d(n) \cdot t$ space. We note that it is open whether $d(n)$-Depth Circuit t-Composition can be solved in polynomial time and $d(n)^c$ space simultaneously for $d(n) > \omega(\log n)$; this is the heart of the NC versus SC question [7, 21].

▶ **Theorem 3.** *For every $\varepsilon \in (0,1)$, $c \geq 1$, $d \in (1,2)$, and $d' < d$, the problem $(\log n)^d$-Depth Circuit n^ε-Composition does not have $\mathsf{SPACE}[(\log n)^{d'}]$-uniform circuits of $n \cdot (\log n)^c$ size and $O((\log n)^{d'})$ depth.*[1]

In other words, we have a problem in $\mathsf{TIME}[n^{1+\varepsilon}] \cap \mathsf{SPACE}[\log^{2-\varepsilon/2}]$ that does not have $\tilde{O}(n)$-size $O(\log^{2-\varepsilon} n)$-depth circuits constructible in $n^{O(\log^{1-\varepsilon} n)}$ time and $O(\log^{2-\varepsilon} n)$ space, for every $\varepsilon \in (0,1)$. Theorem 3 is a significant advance over prior results in the area, which could only prove lower bounds of this form against NP-hard and coNP-hard problems such as SAT and $\overline{\text{SAT}}$ [10, 3], or against non-explicitly given problems in NC [25]. We stress that $O(n)$-size $O(\log n)$-depth lower bounds are in general far more difficult to reason about than one might think: it is open whether every language in $\mathsf{NTIME}[2^n]$ has non-uniform $O(n)$-size $O(\log n)$-depth circuits.

"Easiness Amplification" for Small Circuits

While the problem of proving P does not have $O(n)$-size circuits is notoriously hard, one may try assuming that $\mathsf{P} \subset \mathsf{SIZE}(O(n))$, and record absurd conclusions that follow from it. Might we be able to reach something so absurd that it is provably contradictory? A string of work [17, 12, 25, 8] has established consequences of this form.

---

[1] One might initially believe that Theorem 3 follows quickly from the space hierarchy [26]. It does not: recall the problem $(\log n)^{2-o(1)}$-Depth Circuit n^ε-Composition being lower-bounded is solvable in $n^{1+\varepsilon}$ time, yet we are proving nearly-quadratic space lower bounds for constructing nearly-quadratic depth circuits for it.

The key to the above lower bounds is a lemma which demonstrates how small circuits for these composition problems can be applied to construct small circuits for many more (presumably harder) problems. Let $\mathsf{TISP}[t(n), s(n)]$ denote the class of languages decidable in $t(n)$ time and $s(n)$ space (simultaneously).

▶ **Lemma 4** (Easiness Amplification). *Let $\epsilon > 0$ and let $s(n) \leq \tilde{O}(n)$.*

- *If $s(n)$-IO Circuit n^{ε}-Composition has $\tilde{O}(n)$ size circuits, then every problem in $\mathsf{TISP}[n^{k(n)}, s(n)]$ has $n \cdot (\log n)^{O(k(n)/\epsilon)}$ size circuits, for all constructible functions $k(n) \leq O(\log n / \log\log n)$.*[2]
- *If $s(n)$-IO Circuit n^{ε}-Composition has $n^{1+o(1)}$ size circuits, then for every constant $k \geq 1$, every problem in $\mathsf{TISP}[n^k, s(n)]$ has $n^{1+o(1)}$-size circuits.*

That is, assuming a single problem (solvable in $\tilde{O}(n^{1+\varepsilon})$ time) has nearly-linear-size circuits, we prove that a *much larger* class of problems also has very small circuits (note that $s(n)$-IO Circuit n^{ε}-Composition is in $\mathsf{TISP}[n^{1+\varepsilon}, s(n)]$). We call this phenomenon *easiness amplification*, in contrast with the study of hardness amplification. Let us carefully explain our choice of this term.

In *hardness amplification*, one shows that if a problem from a class $\mathcal{C}$ is "hard" in one sense, one can find another problem from $\mathcal{C}$ that is even "harder." That is, in a hardness amplification theorem, the class of problems being solved remains *about the same*, but the computational lower bound is strengthened in the conclusion (we are amplifying the hardness). However, in what we call easiness amplification, the computational model remains *about the same*, but the class of problems being solved is strictly *increased* in the conclusion: one strictly increases the set of problems which are shown to be easy. To give a specific example and a non-example of easiness amplification, we would say that $\mathsf{NP} \subset \mathsf{P/poly} \Rightarrow \mathsf{PH} \subset \mathsf{P/poly}$ is an easiness amplification, because the notion of "easy" in the conclusion is unchanged from that of the hypothesis. but the set of problems being solved has (probably) strictly increased in the conclusion. On the other hand, $\mathsf{NP} = \mathsf{P} \Rightarrow \mathsf{NEXP} = \mathsf{EXP}$ is not an easiness amplification: the computational model P "blows up" to the larger class EXP.

Lemma 4 says that if the above circuit composition problem (which is in $n^{1+\varepsilon}$ time and $\tilde{O}(n)$ space) had $\tilde{O}(n)$-size circuits, then *every* problem in $n^{o(\log n/\log\log n)}$ time and $\tilde{O}(n)$ space has $n^{1+o(1)}$ size circuits. That is, from a modest speed-up of the circuit composition problem with non-uniform circuits, one obtains an incredible non-uniform simulation of a much larger complexity class. The following is immediate from Lemma 4.

▶ **Corollary 5.** *If n-IO Circuit n^{ε}-Composition has $\tilde{O}(n)$ size circuits, then*

$$\mathsf{TISP}\left[n^{(\log n)/\log\log n}, \tilde{O}(n)\right] \subseteq \mathsf{SIZE}[O(n^2)].$$

By a standard padding argument, we also have $\mathsf{TISP}\left[2^n, 2^{\sqrt{n \log n}}\right] \subseteq \mathsf{SIZE}\left[2^{O(\sqrt{n \log n})}\right]$.

That is, nearly-linear size circuits for circuit composition imply polynomial-size circuits for some problems that are only known to be solvable in *super-polynomial* time, such as detecting a clique of $O(\log n / \log\log n)$ nodes. The second part of Corollary 5 shows that a small improvement on the circuit complexity of a problem solvable in $n^{1+\varepsilon}$ time implies truly *subexponential* circuit upper bounds on *all* problems solvable in 2^n time and $2^{\sqrt{n}}$

[2] As usual in machine-based complexity, one must worry if the functions under consideration are constructible within the resource bounds of the corresponding complexity classes. Throughout the paper, we say a function is "constructible" when it satisfies precisely that condition.

space. For example, the *quantified Boolean formula* problem on formulas of k variables and $2^{\sqrt{k \log k}} \cdot \text{poly}(k)$ size would have a circuit family of $2^{O(\sqrt{k \log k})}$ size. This consequence is in stark contrast to general beliefs regarding exponential time computation, e.g., the Exponential Time Hypothesis of Impagliazzo, Paturi, and Zane [16] which posits that 3-SAT does not have $2^{o(n)}$-time algorithms. Previously, it was only known that $\mathsf{P} \subset \mathsf{SIZE}[O(n)]$ implies $\mathsf{TIME}[2^{O(n)}] \subset \mathsf{SIZE}[2^{o(n)}]$ (by a simple padding argument).

It is interesting to contrast the circuit upper bound consequences of Corollary 5 with the fact that $\mathsf{P} \subset \mathsf{SIZE}[O(n)]$ also implies the negative consequence $\mathsf{P} \neq \mathsf{NP}$ [17]. It would be interesting if $\mathsf{NP} \subset \mathsf{SIZE}[O(n)]$ implied even smaller circuit constructions for PSPACE problems.

Circuit Lower Bounds for SAT

Some of our results extend to the (much harder) SAT problem; in that setting, we can use old proof techniques that are similar to those in prior work on SAT time-space tradeoffs. First we show (Section 4.1) that SAT requires superlogarithmic-space uniform circuits of $\tilde{O}(n)$ size and nearly $\log^2 n$ depth:

▶ **Theorem 6.** *For all $d < 2$, and all $c \geq 1$, SAT $\notin \mathsf{SPACE}[\log^d n]$-uniform $\mathsf{SIZE\text{-}DEPTH}[n \cdot (\log n)^c, (\log n)^d]$.*

With respect to the depth measure, Theorem 6 is an improvement over previous uniform circuit lower bounds for SAT, which established $O(\log n)$-depth limitations (for bounded fan-in and "semi-unbounded" fan-in models). The proof uses the machinery of "alternation-trading proofs" for SAT lower bounds [10, 11, 30], where one assumes a very good SAT algorithm exists, and uses it along with known alternating "speed-up" theorems to derive a contradiction to a known result (generally, some time hierarchy theorem). We show how to use the assumed SAT circuits to obtain a contradiction to the *space* hierarchy theorem, rather than a time hierarchy. This alternate approach leads to stronger results.

Finally, we observe (via old ideas) that the easiness amplification results of Lemma 4 also hold for SAT:

▶ **Theorem 7.** *If SAT has $\tilde{O}(n)$-size circuits, then QBF has $2^{\tilde{O}(\sqrt{n})}$-size circuits.*

This is evidently a new connection between the relative circuit complexity of NP and PSPACE problems. The proof of this theorem can be found in Appendix B.

1.2 Intuition and Comparison

While our lower bounds do apply some ideas from prior work, the proofs of Theorem 2 and 3 have a particular inductive structure that is new to circuit lower bound proofs.[3] The key idea in our lower bounds is encapsulated by the following special case of Lemma 4:

▶ **Lemma 8** (Amplification From Circuit Composition: Special Case)**.** *For every $\varepsilon > 0$, if* Circuit n^{ε}-Composition *has $n^{1+o(1)}$ size circuits, then every problem solvable in* LOGSPACE *has $n^{1+o(1)}$ size circuits.*

Note there are no uniformity assumptions on the circuits in the lemma: the circuits may be arbitrary. This is a strong lower bound amplification result for a particular problem

[3] Readers who doubt this claim are invited to read the "Comparison With Prior Work" below.

in $n^{1+\varepsilon}$ time: small circuits for Circuit n^{ε}-Composition imply small circuits for all of LOGSPACE.

Let us sketch how the lemma is proved. Let M be a machine running in logarithmic space, and assume that small $n^{1+o(1)}$-size circuits exist for Circuit n^{ε}-Composition. Such circuits take two inputs: the description of a smaller circuit C, and an input y to C. We first set the input circuit C to be a $\tilde{O}(n)$-size circuit $C_0(x, \cdot)$ which simulates one step of M on an arbitrary input x of length n: treating the input y as an $O(\log n)$-bit configuration of M on x, $C_0(x, y)$ outputs the configuration y' corresponding to the next step of M on x. Then, $n^{1+o(1)}$-size circuits for Circuit n^{ε}-Composition can be used to construct $n^{1+o(1)}$-size circuits $\{C'_n\}$ which can simulate M on x for about n^{ε} steps, by composing C on the input y for n^{ε} times (using $O(\log n)$ copies, one for each bit of the output configuration).

Now suppose we feed the C'_n circuits as input to Circuit n^{ε}-Composition instead of the C_0 circuits, and repeat the above argument. Note that the C'_n's are only slightly larger than the C_0's. The circuits C'_n (which simulate n^{ε} steps) are being composed for n^{ε} times on the input y. Thus we obtain $n^{1+o(1)}$-size circuits $\{C''_n\}$ which can simulate arbitrary logspace computations for about $n^{2\varepsilon}$ steps. (Please note that this is not obviously true, and our wording here should be taken only as intuition. For example, we have not specified here how to handle arbitrary inputs x of length n.) Once we have the right setup, we can "repeat" the argument for $O(k/\varepsilon)$ times, each time with the new circuit family obtained from the previous iteration plugged in. From this it will follow that n^k-time log-space computations have $n^{1+o(1)}$-size circuits. In particular, given that small-space computations always have small configurations, and assuming we have nearly-linear size circuits that can simulate LOGSPACE for a moderate number of steps, we can concoct new circuits which can simulate LOGSPACE for an arbitrary polynomial number of steps, while keeping the circuit size around $n^{1+o(1)}$.

To prove the main circuit lower bound of Theorem 2, we start by assuming that General Circuit n^{ε}-Composition has logspace-uniform $n^{1+o(1)}$-size circuits, and wish to derive a contradiction. First we note that General Circuit n^{ε}-Composition is "complete" for $n^{1+\varepsilon}$ time in a certain precise sense, and thus cannot be solved faster than this bound. Using the argument of the above paragraph, we derive that every language in LOGSPACE also has almost-linear-size circuits. But if every language in LOGSPACE has small circuits, it can be shown that every logspace-uniform circuit family can be produced by *very small* circuits, by a "de-padding"' trick of Santhanam and Williams [25] which gives the uniform algorithm much smaller inputs. Indeed, assuming LOGSPACE has almost-linear-size circuits, it can be shown that every problem with small logspace-uniform circuits can also be decided very efficiently, with only $o(n)$ bits of advice. We can use this consequence to prove that General Circuit n^{ε}-Composition is solvable so efficiently that it contradicts our earlier time lower bound for the problem.

Note that if the original circuits for Circuit n^{ε}-Composition are assumed to be uniform (or of low depth, respectively) then the composition circuits in the above iterated constructed are also uniform (or of low depth, respectively). The main goal in the proof of the depth lower bound (Theorem 3) is to extend the above amplification lemma to simulate all problems in SPACE$[(\log n)^d]$ with uniform circuits of depth only $o(\log n)^d$. But such circuits can always be evaluated in SPACE$[o((\log n)^d)]$, so this consequence contradicts the space hierarchy.

We observe that (in contrast to the techniques of Santhanam and Williams) the proof technique in the amplification lemma looks to be inherently *non-relativizing*. The first circuit in our composition only simulates a logspace machine for one step, and our composition problem only simulates a given circuit for some n^{ε} steps, crucially using the fact that the output of the circuit (the configuration size) is only $n^{o(1)}$ bits at every stage of the induction.

Informally, there is no "room" in our circuit simulation of LOGSPACE to write down long oracle queries. To confirm this intuition, we construct an oracle relative to which our main lower bound is false, in Appendix A.

The analogous lower bound results for SAT follow from the fact that the circuit composition problem can also be solved using alternations rather than computing it serially: one can simply (existentially) guess the intermediate values output in the circuit composition, and (universally) verify the outputs in parallel. If SAT has nearly-linear-size circuits, then Σ_kSAT also has nearly-linear-size circuits, which also allows for a very efficient circuit simulation of small-space computation.

Comparison With Prior Arguments

We argue that the above proof technique (behind Lemma 4, Theorem 2, and Theorem 3) is a fundamentally different way to derive an efficient simulation of small-space computation, compared to earlier arguments.

1. **SAT Time-Space Lower Bounds.** In the SAT time-space lower bounds based on extending Savitch's theorem (such as Fortnow-Lipton-Van Melkebeek-Viglas [10, 11]), the use of *alternating computation* is critical: generalizing Savitch's Theorem, one constructs a simulation which "guesses" a few configurations of a small-space machine that will be visited later in the computation, then "universally verifies" the guesses independently. In this way, the simulation problem for a small-space computation is partitioned into small parts which can then be solved independently (in parallel). The proof of the Easiness Amplification Lemma (Lemma 4) uses no alternation or parallelism. Indeed, Lemma 4 shows how the *serial* process of simulating a small-space computation for t steps can be self-improved by assuming that $n^{1+\varepsilon}$ steps can be simulated with $n^{1+o(1)}$ -size circuits, and repeatedly feeding "small circuit copies" into circuits which perform this efficient step simulation. Note that in our later lower bounds for SAT, we do use alternations, showing that this methodology can derive similar results but under a seemingly stronger assumption (namely, the assumption "SAT has small circuits" instead of "circuit composition has small circuits").
2. **Hardness Amplification From Self-Reducibility.** The proof technique here is also different from earlier arguments based on the self-reducibility of a problem (such as those found in Lipton-Viglas, Allender-Koucky, and Lipton-Williams [18, 2, 19]). In those arguments, one decomposes a given computation into disjoint "parts", applies an assumed "good" simulation to each part separately, then proves that the new overall simulation is now similarly "good." The results proven in these papers are hardness amplifications: given a "pretty good" simulation of some particular kind of resource-bounded computation, we can obtain an extremely efficient simulation of the same bounded computation. (For example, if the NC^1-complete formula evaluation problem has TC^0 circuits of polynomial size, then it also has TC^0 circuits of $n^{1.0001}$ size [2]. Thus a weak TC^0 size lower bound against this NC^1 problem would separate NC^1 from TC^0.) Our results have a different character: we show that if one can simulate ultra-efficient computations with small circuits, then one can simulate all "pretty good" computations with small circuits. *We are improving the class of computations being simulated, instead of improving the simulation itself.* Again, this is the difference between hardness amplification and what we call "easiness amplification."
3. **Top-Down Versus Bottom-Up.** In Lemma 4, we begin with a small circuit that simulates one step of the computation correctly, and apply the hypothesis in a way that lets us build a (slightly larger) circuit simulating n^{ε} steps correctly. From that circuit, and our

hypothesis, we build another (slightly larger) circuit simulating $n^{2\varepsilon}$ steps correctly, and so on, until we have obtained a small circuit which can simulate n^k steps correctly for any desired k. Methodologically, this is a "bottom-up" way of simulating small-space computations faster: building up small circuits for simulating long running times, starting with a trivial circuit for simulating one step.
This should be contrasted with the "top-down" approach of extensions of Savitch's Theorem in item 1 above, where one guesses a way to partition the computation into pieces, then verifies the pieces recursively. The arguments based on self-reducibility in item 2 above also have a "top-down" form: they start by decomposing the entire computation into small parts, then substitute small copies of an improved circuit in place of each of the parts. Because we view the machine simulation problem in a bottom-up way, this allows us to prove a lower bound on a much simpler problem (a circuit composition problem solvable in $n^{1+\varepsilon}$ time) compared to earlier unconditional lower bounds of this type (for SAT and for QBF).

The most canonical results in the same spirit of our work are classical theorems such as $\mathsf{NP} \subset \mathsf{P/poly} \Rightarrow \mathsf{PH} \subset \mathsf{P/poly}$, where one replaces the quantifiers in a Σ_k computation by larger circuits. Of course there are other obvious differences there: in our setting, we want to keep the circuit size basically fixed (around $n^{1+o(1)}$) and we want to increase the *running times* of the problems we can solve with such circuits, in each inductive step. As far as we can tell, we need to be simulating small-space computations to pull off this kind of easiness amplification. It would be highly desirable to remove the small-space restriction in these lower bounds.

2 Preliminaries

We assume basic familiarity with concepts in complexity theory [4]. Below are some definitions and notions specific to this paper.

▶ **Definition 9.** A language L is in the class $\mathsf{TISP}[t(n), s(n)]$ if it can be decided in $O(t(n))$ time and $O(s(n))$ space simultaneously on a multitape Turing machine.

In our results on the SAT problem, we also use standard notions of alternating Turing machines:

▶ **Definition 10.** A language L is in the class $\Sigma_{a(n)}\mathsf{TISP}[t(n), s(n)]$ if it can be decided in $O(t(n))$ time and $O(s(n))$ space simultaneously on a multitape Turing machine using at most $a(n)$ alternations.

Sometimes we say "algorithm" instead of "multitape Turing machine." These should be thought of as synonymous. We need the multitape model to ensure that our algorithmic computational model has an efficient translation to small-size circuits.

In the following, let $\mathcal{C}$ be any time or space complexity class (such as $\mathsf{TIME}[n^2]$, P, $\mathsf{SPACE}[\log^2 n]$, etc.). For a circuit C, let $|C|$ be the length of its description in binary.

▶ **Definition 11.** A $\mathcal{C}$-uniform circuit family $\{C_n\}$ has the property that there is a multitape Turing machine A implementable in $\mathcal{C}$, such that for every n, $A(1^n)$ prints the $|C_n|$-length description of C_n in binary. (Strictly speaking, as the class $\mathcal{C}$ consists of decision problems, A should output only one bit. This is easily accommodated by requiring for all n and $i = 1, \ldots, |C_n|$ that $A(1^n, i)$ outputs the ith bit of the description of C_n.)

For example, a P-uniform circuit family comes with a polynomial-time multitape Turing machine A which on 1^n prints the nth circuit in the family.

For a language $L \subseteq \{0,1\}^\star$, define $L_n = L \cap \{0,1\}^n$ to be the strings in L of length n. A language L is in $\mathsf{SIZE}[s(n)]$ if for every $n \geq 0$, there is a circuit of at most $s(n)$ gates that computes L_n. L is in $\mathsf{DEPTH}[d(n)]$ if for every $n \geq 0$ there is a circuit that computes L_n such that the longest path from source to sink in this circuit has length at most $d(n)$. L is in $\mathsf{SIZE\text{-}DEPTH}[s(n), d(n)]$ if for every $n \geq 0$ there is a circuit with at most $s(n)$ gates computing L_n such that the longest path in the circuit has length at most $d(n)$. For example, the class NC^1 can be rewritten as $\mathsf{SIZE\text{-}DEPTH}[n^{O(1)}, O(\log n)]$. The following observation is useful in our depth lower bounds:

▶ **Proposition 12.** $\mathsf{SPACE}[s(n)]$-*uniform* $\mathsf{DEPTH}[s(n)] \subseteq \mathsf{SPACE}[s(n)]$.

Proof. For a circuit family $\{C_n\}$ that is $s(n)$-space uniform, on an n-bit input we can always use $O(s(n))$ space to generate any gate information of the circuit C_n necessary for a simulation. Because C_n also has $s(n)$ depth, we only require $O(s(n))$ additional space to simulate it (for a reference, see Vollmer [29]). ◀

Generalized Circuit Composition

In the following, let $Z : \{0,1\}^k \to \{0,1\}^k$ be a function that always returns 0^k.

▶ **Definition 13.** In the General Circuit t-Composition problem, we are given:

- A k-bit input x.
- A circuit C_{in} of size n over the basis $\{AND, OR, NOT\}$ implementing a function from $3k$ bits to k bits (where k can range up to n).
- A circuit C_{out} of size t over the basis $\{C_{in}, Z\}$, implementing a function from k bits to k bits. In particular, C_{out} is presented as a DAG, where every node has indegree 0, 1, or 3; every edge of C_{out} carries a k-bit string.
- An integer j in $1, \ldots, k$.

The input is accepted iff the jth bit printed by $C_{out}(x)$ is 1.

That is, every node in C_{out} of indegree 3 implements C_{in}, taking in $3k$ bits and outputting k bits.) We need indegree 3 in order to carry out a multitape TM simulation effectively.)

In our lower bound proofs, we only require two key properties of General Circuit t-Composition.

1. The first is that n-IO Circuit t-Composition for a circuit C is a special case of the General Circuit-t-Composition Problem. This is true because n-IO Circuit t-Composition corresponds to the case where C_{out} is simply a straight line of t copies of C_{in}, where $C_{in}(x, 0^k, 0^k) = C(x)$ for all x.
2. The second property is that General Circuit t-Composition is essentially complete for $\tilde{O}(nt)$ time, as the below theorem demonstrates.

▶ **Theorem 14.** *Let $L \in \mathsf{TIME}[n^{1+\varepsilon}]$. L can be reduced in $\tilde{O}(n)$ time to* General Circuit n^ε-Composition.

Proof (Sketch). Let M be multitape Turing machine for L. We follow the proof of the Size Reduction Lemma (Lemma 3.2) in Lipton and Williams [19], which shows how to "decompose" an arbitrary time $n^{1+\varepsilon}$ computation into circuits of size n^ε in which every gate takes a constant number of $O(n)$-bit "blocks" of input, simulates an $O(n)$-time machine M' on the blocks, and outputs an $O(n)$-bit block.

In particular, they convert M into an equivalent two-tape *oblivious* M' (where, for every n and input x of length n, the tape head movements of $M'(x)$ depend only on n). M' runs in $t = \tilde{O}(n^{1+\varepsilon})$ time, via the Hennie-Stearns simulation [14]. This oblivious two-tape simulation is polylog-time uniform, in that we can determine the head positions in any given step $i = 1, \ldots, t$ in $\text{poly}(\log t)$ time.

Next, M' is made *block-respecting* in the sense of Hopcroft-Paul-Valiant [15]. This is a machine M'' running in $t'(n) = O(t(n))$ time, whose computation can be neatly partitioned into $\frac{t'(n)}{b(n)}$ time blocks of $O(b(n))$ steps each, and each tape is partitioned into $O(\frac{t'(n)}{b(n)})$ tape blocks of $O(b(n))$ cells each, for any constructible $b(n) \leq t(n)$. We set $b(n) = O(n)$. For each time block, and each tape head, the head stays within exactly one tape block. Therefore each time block can be viewed as running on an input of $O(b(n))$ bits, and each block outputs $O(b(n))$ bits (the new content of those tape blocks). The Hopcroft-Paul-Valiant simulation maintains the obliviousness of M'.

For our generalized circuit composition instance, the circuit C_{in} simply simulates a time block of length $b(n)$, assuming the initial input is of length n. It takes $O(n)$ bits and outputs $O(n)$ bits, having $\tilde{O}(n)$ size. The circuit C_{out} connects these n^{ε} time blocks together: each gate of C_{out} implements a time block. C_{out} is built by determining the head movements of the oblivious M'' spaced n^{ε} steps apart, and wiring together time blocks that share common tape blocks (or adjacent time blocks that share adjacent tape blocks, depending on the head position). By design, the resulting circuit C_{out} is equivalent to the original Turing machine M on n-bit inputs. ◀

As a corollary, there is a constant $c > 0$ such that General Circuit n^{ε}-Composition needs at least $n^{1+\varepsilon}/(\log n)^c$ time, by the time hierarchy theorem.

2.1 Simulating Bounded Space

Many results we obtain stem from composing circuits that simulate space-bounded computations. The following notion is useful:

▶ **Definition 15.** Fix a machine M, and an input length n. The simulation machine $Sim_{t(n)}(x, c, i)$ takes a string x of length n, configuration c of M on x, and an index bit i, simulates $M(x)$ from configuration c for $t(n)$ steps, then outputs the ith bit of the resulting configuration c'.

For machines M running in space $s(n)$, $Sim_{t(n)}$ has circuits of size $\tilde{O}(t(n) \cdot (n + s(n)))$. In our amplification lemma (Lemma 4), we construct much more efficient circuits, assuming circuit composition has small circuits.

3 LOGSPACE-Uniform Circuit Lower Bounds

We begin this section with our amplification lemma, which is used to prove most of the following results.

▶ **Reminder of Lemma 4.** *Let $\epsilon > 0$ and let $s(n) \leq \tilde{O}(n)$.*

- *If $s(n)$-IO Circuit n^{ε}-Composition has $\tilde{O}(n)$ size circuits, then every problem in $\mathsf{TISP}[n^{k(n)}, s(n)]$ has $n \cdot (\log n)^{O(k(n)/\epsilon)}$ size circuits, for all constructible functions $k(n) \leq O(\log n / \log\log n)$.*
- *If $s(n)$-IO Circuit n^{ε}-Composition has $n^{1+o(1)}$ size circuits, then for every constant $k \geq 1$, every problem in $\mathsf{TISP}[n^k, s(n)]$ has $n^{1+o(1)}$-size circuits.*

Proof. We start by proving the first bullet. Assume $s(n)$-IO Circuit n^ε-Composition has $\tilde{O}(n)$ size circuits. Recall that $s(n)$-IO Circuit n^ε-Composition takes as input a circuit C with $s(n)$ inputs and $s(n)$ outputs, a string $y \in \{0,1\}^k$, and an index i, then prints the ith bit of the n^ε-fold composition of C on y.

By assumption, $s(n)$-IO Circuit n^ε-Composition has a circuit family $\{E_n\}$ of $O(n \cdot (\log n)^d)$ size.

Now take an arbitrary $L \in \mathsf{TISP}[n^{k(n)}, s(n)]$ for some function $k(n) \leq O(\log n / \log\log n)$, and let M be a machine recognizing L in $n^{k(n)}$ time and $s(n)$ space. We will show that L has circuits of size $n \cdot (\log n)^{O(k(n)/\varepsilon)}$, by constructing the circuit inductively.

For the base case, let $M_0(x, c, i) = Sim_1$ (from Definition 15) be a machine which simulates M on x from the configuration c for one step, then outputs the ith bit of the next configuration. This simulation can easily be done in linear time, by first looking up the input bit read by the input head, simulating one step of the computation, then outputting the ith bit of the resulting configuration. By the usual conversion of algorithms into circuits, there is a circuit family that is equivalent to Sim_1 with $n \cdot (\log n)^a$ size for some constant a. Let $C_0(x, c, i)$ denote a generic circuit from this family. We choose a convention for expressing the description of C_0 such that, given a bit string D which is the description of C_0, and given an n-bit input x, a description D_x of the circuit $C_0(x, \cdot)$ (i.e., the first n inputs of C_0 are filled in with the bits of x) is obtained by substituting the n bits of x into n particular bit positions $i_1, \ldots, i_n$ of the description D. Such a description only takes $O(z \log z)$ bits to describe, where z the circuit size.

Now fix $b \geq 0$. Suppose we have constructed a circuit $C_b(x, c, i) = Sim_{n^{\varepsilon \cdot b}}$ of size $n \cdot (\log n)^{a+b\cdot(d+1)}$ that simulates $n^{\varepsilon \cdot b}$ steps of the machine M on x of length n. Consider the circuit

$$C_{b+1}(x, c, i) := E_{n\cdot(\log n)^{a+b\cdot(d+1)+1}}(C_b(x, \cdot), c, i)\,.$$

(Note that by our convention for encoding circuits, $C_b(x, \cdot)$ should be construed as a bit string but with n bit positions that are unassigned free variables.) Since the language $s(n)$-IO Circuit n^ε-Composition simulates $n^{\varepsilon \cdot b}$ steps of M with each evaluation of the circuit C_b, and the circuit C_b is being composed for n^ε times, $C_{b+1}(x, c, i)$ simulates $n^{\varepsilon \cdot b} \cdot n^\varepsilon = n^{\varepsilon \cdot (b+1)}$ steps of M on x.

Furthermore, since the circuit C_b is of size $O(n \cdot (\log n)^{a+b\cdot(d+1)})$, the binary representation of C_b has length $\ell = O(n \cdot (\log n)^{a+b\cdot(d+1)+1})$. Therefore the input to the circuit C_{b+1} has length $O(\ell)$, and the size of the circuit C_{b+1} would then be of size

$$O(\ell \cdot (\log \ell)^d) \leq O(n \cdot (\log n)^{a+b\cdot(d+1)+1} \cdot (\log(n(\log n)^{a+b\cdot(d+1)+1})^d)).$$

For $b < \log n / \log\log n$, we have $n \log^{a+b\cdot d} n \leq n^{1+d} \log^a n$; in that case, the size bound can be simplified to

$$O(n \cdot (\log^{a+b\cdot(d+1)+1} n) \cdot (\log n)^d) = O(n \log^{a+b\cdot(d+1)+(d+1)} n) = O(n \log^{a+(b+1)\cdot(d+1)} n).$$

We have shown that given a circuit C_b of size $O(n \cdot (\log n)^{a+b\cdot(d+1)})$ that simulates $n^{\varepsilon \cdot b}$ steps of M on x, we can construct a circuit C_{b+1} of size $O(n \log^{a+(b+1)\cdot(d+1)})$ that simulates $n^{\varepsilon\cdot(b+1)}$ steps of M on x. Therefore for every $b \leq o(\log n / \log\log n)$, there is a circuit C_b of size $n \cdot (\log n)^{O(b)}$ that simulates $n^{\varepsilon \cdot b}$ steps of the space-$s(n)$ machine M. Setting $b = k(n)/\varepsilon$ and c to be the initial configuration of M on inputs of length n, we obtain a circuit $C_{k(n)/\varepsilon}$ of size $n \cdot (\log n)^{O(k(n)/\varepsilon)}$ that can simulate the entire computation of M and output the final configuration of $M(x)$, which can then be used to decide L. Since L was arbitrary, we

conclude that

$$\mathsf{TISP}[n^{k(n)}, s(n)] \subseteq \mathsf{SIZE}[n \cdot (\log n)^{O(k(n)/\epsilon)}],$$

which completes the proof of the first bullet.

To prove the second bullet, let $\varepsilon \in (0,1)$ and $k \geq 1$ be constant. We run the same argument on an arbitrary machine M using n^k-time and $s(n)$-space, but instead we assume there are $n^{1+1/f(n)}$-size circuits for the n^ε-composition problem, where $f(n)$ is an unbounded function. For every constant $b \geq 0$, tracking the growth of C_b in the above argument, we obtain a circuit $C_b(x,c,i)$ of size at most $n^{(1+1/f(n))^{db}}$ (for some universal constant $d > 0$) for simulating $n^{\varepsilon b}$ steps of M on an input of length n. By setting $b := k/\varepsilon$ as in the previous case, the resulting circuit C_b can then simulate M entirely on all inputs of length n. We can define an unbounded function $g : \mathbb{N} \to \mathbb{N}$ such that

$$(1 + 1/f(n))^{\log f(n)} \leq 1 + 1/g(n)$$

for all n. Then for all constants b and for all sufficiently large n, the size of C_b is $n^{(1+1/f(n))^{db}} \leq n^{(1+1/f(n))^{\log f(n)}} \leq n^{1+1/g(n)} \leq n^{1+o(1)}$. ◀

One property of note in the above proof is that uniformity can be applied to the circuits, without changing the argument. For example, if the small circuits for n^ε-circuit composition are LOGSPACE-uniform, then the small circuits for $\mathsf{TISP}[n^k, s(n)]$ are also LOGSPACE-uniform, assuming that k is constant (the argument becomes more complicated when adding an unbounded number of iterations).

▶ **Reminder of Theorem 2.** *For $0 < \varepsilon < 1$, the decision problem* General Circuit n^ε-Composition *does not have* LOGSPACE*-uniform $n^{1+o(1)}$ size circuits.*

Proof. Assume there is an $\varepsilon > 0$ such that General Circuit n^ε-Composition has LOGSPACE-uniform circuits of $n^{1+o(1)}$ size. Since Circuit n^ε-Composition is a special case of the general problem, it must also have such circuits. Therefore by Lemma 4, our assumption implies that for every constant $c \geq 1$, every problem in $\mathsf{TISP}[n^c, O(\log n)]$ has $n^{1+o(1)}$-size circuits as well. That is, we have

$$\mathsf{LOGSPACE} \subset \mathsf{SIZE}[n^{1+o(1)}]. \tag{1}$$

Since the circuits for General Circuit n^ε-Composition are LOGSPACE-uniform, there is an $O(\log n)$-space algorithm A that on input 1^n prints a $n^{1+o(1)}$-size circuit C_n computing General Circuit n^ε-Composition on n-bit instances. We are going to prove that General Circuit n^ε-Composition can be simulated in $n^{1+\varepsilon/2}$ time with about $n^{\varepsilon/2}$ bits of advice, implying a contradiction.

Similar to Santhanam and Williams [25], our next move is to define, for every rational $\alpha \in (0,1)$, a padded language $L_\alpha = \{(1^{n^\alpha}, n, i) \mid$ the ith bit of the circuit printed by $A(1^n)$ is $1\}$.

When $\alpha \in (0,1)$ is a fixed constant, we note the following properties of L_α:

(a) L_α is in LOGSPACE: on an m-bit instance $(1^{n^\alpha}, n, i)$, a machine deciding L_α only has to simulate A on 1^n in $O((\log m)/\alpha)$ space, and maintain a $(1 + o(1)) \log n$-bit counter for i, until the ith output bit of $A(1^n)$ is printed. (Note that $(1 + o(1)) \log n \leq O(\log m)$.) Hence by (1), L_α has an $n^{1+o(1)}$-size circuit family $\{D_m\}$, for every $\alpha > 0$.

(b) For an integer $n > 0$, if we want to know bits of the circuit printed by $A(1^n)$, the length of a relevant instance of L_α is only $|(1^{n^\alpha}, n, i)| \leq O(n^\alpha)$. Let $m(n)$ be the length of such an instance. On $m(n)$-bit instances, L_α outputs bits describing an $n^{1+o(1)}$-size circuit C_n which in turn solves n-bit instances of General Circuit n^ε-Composition.

Let $\alpha = \varepsilon/2$. We can decide General Circuit n^ε-Composition in $\tilde{O}(n^{1+\varepsilon/2})$ time with only $n^{\varepsilon/2+o(1)}$ bits of advice, as follows. On n-bit instances of the problem, our advice string is a description of the circuit $D_{m(n)}$ of size $m(n)^{1+o(1)}$ for $L_{\varepsilon/2}$ from item (a) above. From item (b), the length of the advice string is $m(n)^{1+o(1)} \leq n^{\varepsilon/2+o(1)}$. Given $D_{m(n)}$, our machine for circuit composition will evaluate $D_{m(n)}$ on $(1^{n^\alpha}, n, i)$ for all $i = 1, \ldots, n^{1+o(1)}$. This evaluation will, by definition, generate a description of an $n^{1+o(1)}$-size circuit C_n that solves n-bit instances of our problem. Sending the n-bit input to C_n, we can decide General Circuit n^ε-Composition in time $n^{1+o(1)} \cdot m(n)^{1+o(1)} \leq n^{1+\varepsilon/2+o(1)}$, with $n^{\varepsilon/2+o(1)}$ bits of advice.

However, since General Circuit n^ε-Composition is hard for $\mathsf{TIME}[n^{1+\varepsilon}]$ under $\tilde{O}(n)$-time reductions (Lemma 14), it follows from our simulation that *every* language in $\mathsf{TIME}[n^{1+\varepsilon}]$ is contained in $\mathsf{TIME}[n^{1+\varepsilon/2+o(1)}]/n^{\varepsilon/2+o(1)}$. This contradicts the time hierarchy theorem with sub-linear advice (which is folklore; see [25] for a proof). ◀

▶ **Reminder of Corollary 5.** *If n-IO Circuit n^ε-Composition has $\tilde{O}(n)$ size circuits, then*

$$\mathsf{TISP}\left[n^{(\log n)/\log\log n}, \tilde{O}(n)\right] \subseteq \mathsf{SIZE}[O(n^2)].$$

By a standard padding argument, we also have $\mathsf{TISP}\left[2^n, 2^{\sqrt{n \log n}}\right] \subseteq \mathsf{SIZE}\left[2^{O(\sqrt{n \log n})}\right]$.

Proof. We wish to maximize the time bound $n^{k(n)}$ in the consequence of Lemma 4 such that the hypothesis implies $O(n^2)$-size circuits for that time bound. This can be done by setting $n = (\log n)^{c_1 \cdot k(n)/\varepsilon}$ for a constant $c_1 > 0$. Solving for $k(n)$, we find $k(n) = c_2 \cdot \varepsilon(\log n)/(\log\log n)$ for a constant $c_2 > 0$. By Lemma 4, when $s(n)$-IO Circuit n^ε-Composition has $\tilde{O}(n)$ size circuits we have $\mathsf{TISP}[2^{c\varepsilon(\log n)^2/\log\log n}, O(n)] \subseteq \mathsf{SIZE}[n^2]$. By padding the input by $n \mapsto 2^{O(\sqrt{n \log n})}$, we also conclude that $\mathsf{TISP}[2^n, 2^{O(\sqrt{n \log n})}] \subseteq \mathsf{SIZE}[2^{O(\sqrt{n \log n})}]$. ◀

4 Log-Depth Circuit Lower Bounds

We now turn to proving uniform lower bounds for composing circuits of low depth. We can also prove an amplification lemma in this regime:

▶ **Lemma 16.** *Let $\varepsilon > 0$, $c, d \geq 1$, and $e \in [1, d)$. Let $k(n) = o(\log n/\log\log n)$ be constructible. If $(\log n)^d$-Depth Circuit n^ε-Composition has $\mathsf{SPACE}[(\log n)^e]$-uniform circuits of $n \cdot (\log n)^c$ size and $O((\log n)^e)$ depth, then every problem in $\mathsf{TISP}[n^{k(n)}, (\log n)^d]$ has $\mathsf{SPACE}[(\log n)^e]$-uniform circuits of $n \cdot (\log n)^{(c+1)\cdot k(n)/\varepsilon}$ size and $O((\log n)^e)$ depth.*

Proof. The proof is similar to Lemma 4. If we assume that $(\log n)^d$-Depth Circuit n^ε-Composition has $\mathsf{SPACE}[(\log n)^e]$-uniform circuits of $n \cdot (\log n)^c$ size and $O((\log n)^e$ depth, then we know that $(\log n)^d$-Depth Circuit n^ε-Composition can *compose its own circuit* in the same way that Circuit n^ε-Composition can, since the depth of the circuit is $(\log n)^e = o((\log n)^d)$.

As a result, if one step of a $\mathsf{SPACE}[s(n)]$ computation can be simulated with a $O(n)$-size $(\log n)^d$-depth circuit, then n^ε steps can be simulated with a $O(n \cdot (\log n)^{c+1})$ size circuit, $n^{2\varepsilon}$ steps can be simulated with a $O(n \cdot (\log n)^{2\cdot(c+1)})$ size circuit, and so on, using the $(\log n)^d$-Depth Circuit n^ε-Composition circuits. Since these are constructed by simply composing $(\log n)^d$-Depth Circuit n^ε-Composition circuits, if the original is $\mathsf{SPACE}[(\log n)^e]$-uniform and $O((\log n)^e)$ depth then all of the constructed circuits will be as well, as long as the circuit size remains polynomial-sized. By composing $k(n)/\varepsilon$ times,

we can construct $\mathsf{SPACE}[(\log n)^e]$-uniform $n \cdot (\log n)^{O(k(n)/\varepsilon)}$-size $O(\log n)^e$-depth circuits for $\mathsf{TISP}[n^{k(n)}, s(n)]$. Setting $s(n) = (\log n)^d$ completes the proof. ◀

▶ **Reminder of Theorem 3.** *For every $\varepsilon \in (0,1)$, $c \geq 1$, $d \in (1,2)$, and $e < d$, the problem $(\log n)^d$-Depth Circuit n^ε-Composition does not have $\mathsf{SPACE}[(\log n)^e]$-uniform circuits of $n \cdot (\log n)^c$ size and $O((\log n)^e)$ depth.*

Proof. Assume there are $\varepsilon > 0$, $d \in (1,2)$, and $e < d$ that $(\log n)^d$-Depth Circuit n^ε-Composition has $\mathsf{SPACE}[(\log n)^e]$-uniform circuits of $n \cdot (\log n)^c$ size and $O((\log n)^e)$ depth. Setting $k(n) := (\log n)^{d-1}$, we have $d - 1 < 1$ by assumption, therefore $k(n) = o(\log n / \log\log n)$. Applying Lemma 16 to our assumption, we can infer that the class $\mathsf{SPACE}[(\log n)^d] = \mathsf{TISP}[n^{O((\log n)^{d-1})}, (\log n)^d]$ has $\mathsf{SPACE}[(\log n)^e]$-uniform circuits of $n \cdot (\log n)^c$ size and $O((\log n)^e)$ depth. That is, every problem in $\mathsf{SPACE}[(\log n)^d]$ also has $\mathsf{SPACE}[(\log n)^e]$-uniform circuits of $O((\log n)^e)$ depth. But by Proposition 12,

$$\mathsf{SPACE}[(\log n)^e]\text{-uniform } \mathsf{DEPTH}[(\log n)^e] \subseteq \mathsf{SPACE}[(\log n)^e],$$

so we have actually derived $\mathsf{SPACE}[(\log n)^d] \subseteq \mathsf{SPACE}[(\log n)^e]$. This contradicts the space hierarchy theorem [27], because $e < d$. ◀

4.1 Depth Lower Bound for SAT

Now we show that the SAT problem does not have subquadratic-space-uniform $\tilde{O}(n)$-size circuits of $\log^{2-\varepsilon} n$ depth, for all $\varepsilon > 0$. The results here will take the typical form of "alternation-trading proofs" [31], where one quickly simulates a space-bounded computation with alternations, removes the alternations using efficient (assumed) SAT circuits, and attempts to prove a contradiction. The key difference between our proof approach and prior ones is that we are able to use the space hierarchy theorem to establish the contradiction, which leads to a stronger space and depth lower bound.

We will use the following powerful simulation lemma of Reischuk (based on Nepomnjascii [20]) in our proof.

▶ **Lemma 17** ([23], p.282). *For constructible functions $t_1(n)$, $t_2(n)$, $s(n)$ and $a(n)$, we have the containment*

$$\Sigma_{a(n)}\mathsf{TISP}[t_1(n)^{t_2(n)}, s(n)] \subseteq \Sigma_{a(n)+t_2(n)}\mathsf{TISP}[a(n) \cdot s(n) + t_1(n) \cdot t_2(n) \cdot s(n), t_1(n) \cdot s(n)].$$

In general, Reischuk's lemma shows how alternating computations with large time bounds and small space bounds can be converted into alternating computations with much lower time bounds and slightly larger space bounds.

We need another lemma showing how good SAT circuits can yield circuits for alternating computations. It is similar to other arguments of this kind, but we include it for completeness:

▶ **Lemma 18.** *Let $c, d > 0$. If $SAT \in \mathsf{SPACE}[(\log n)^d]$-uniform $\mathsf{SIZE\text{-}DEPTH}[n \cdot (\log n)^c, (\log n)^d]$, then for $k \geq 1$*

$$\Sigma_k\mathsf{TIME}[n] \subseteq \mathsf{SPACE}[(\log n)^d]\text{-uniform } \mathsf{SIZE\text{-}DEPTH}[n \cdot (\log n)^{(c+1)\cdot k}, (\log n)^d].$$

Proof. The proof is by induction on k, the number of alternations. For $k = 1$, the statement becomes $\mathsf{NTIME}[n] \subseteq \mathsf{SPACE}[(\log n)^d]$-uniform $\mathsf{SIZE\text{-}DEPTH}[n \cdot (\log n)^{c+1}, (\log n)^d]$, which is true by assumption.

Suppose that for some fixed value of k the lemma holds. Consider some $L \in \Sigma_{k+1}\mathsf{TIME}[n]$. We can then construct a circuit of size $n \cdot (\log n)^{(c+1)\cdot(k+1)}$ and depth $O((\log n)^d)$ using $O((\log n)^d)$ space that computes L. Since L is verified in linear time, the number of bits of nondeterminism in the first alternation is at most linear, so there is a $\Pi_k\mathsf{TIME}[n]$ verifier $V(x, y)$ for L that takes both the input x to L and the first set of nondeterministic bits y as input and checks whether y is a valid witness that $x \in L$. By assumption, V has $\mathsf{SPACE}[(\log n)^d]$-uniform $\mathsf{SIZE\text{-}DEPTH}[n \cdot (\log n)^{(c+1)\cdot k}, (\log n)^d$ circuits.

Consider the Circuit-SAT instance that consists of the above circuit that computes V as well as the input x that is meant to be the input of the original language L. The description of this circuit is of size $N = n \cdot (\log n)^{(c+1)\cdot k+1}$, so by assumption there is a circuit of size $N \cdot (\log N)^c = n \cdot (\log n)^{(c+1)\cdot k+1} \cdot (O(\log n))^c = n \cdot (\log n)^{(c+1)\cdot(k+1)}$ and depth $O((\log N)^d) = O((\log n)^d)$ computable in $\mathsf{SPACE}[(\log N)^d] = \mathsf{SPACE}[(\log n)^d]$ that solves this SAT instance.

Both the Π_k verifier circuit for L and the Circuit-SAT circuit can be constructed in $\mathsf{SPACE}[(\log n)^d]$, and by hard-coding the description of the verifier circuit as input to the Circuit-SAT instance, the resulting circuit will solve L. Furthermore the size and depth of the circuit is simply the size and depth of the Circuit-SAT instance, since the Π_k circuit is fed as a description (only the size of the description matters). This circuit then exists for every $L \in \Sigma_{k+1}\mathsf{TIME}[n]$. Therefore if

$$\Sigma_k\mathsf{TIME}[n] \subseteq \mathsf{SPACE}[(\log n)^d]\text{-uniform } \mathsf{SIZE\text{-}DEPTH}[n \cdot (\log n)^{(c+1)\cdot k}, (\log n)^d]$$

then

$$\Sigma_{k+1}\mathsf{TIME}[n] \subseteq \mathsf{SPACE}[(\log n)^d]\text{-uniform } \mathsf{SIZE\text{-}DEPTH}[n \cdot (\log n)^{(c+1)\cdot(k+1)}, (\log n)^d]$$

and the induction holds for all $k \geq 1$.

Note that the above argument holds even if the circuits are non-uniform. Furthermore, as long as the size of the circuits remain fairly small, k can also be a function of the input size. If $k(n) = o(\log n/\log\log n)$, then the circuits produced will still have $n^{1+o(1)}$-length descriptions at every step of the induction, since

$$n \cdot (\log n)^{(c+1)\cdot o(\log n/\log\log n)} \leq n \cdot n^{o(c+1)} = n^{1+o(1)}.$$

The main factor that determines the size, depth and uniformity of the next alternation in the induction is the size of the previous circuit in the induction. Therefore, as long as that circuit has $\tilde{O}(n)$ size, the inductive step will hold. ◀

▶ **Reminder of Theorem 6.** *For all $d < 2$, and all $c \geq 1$,*

$$SAT \notin \mathsf{SPACE}[\log^d n]\text{-}uniform\ \mathsf{SIZE\text{-}DEPTH}[n \cdot (\log n)^c, (\log n)^d]\,.$$

Proof. Assume that SAT $\in \mathsf{SPACE}[(\log n)^d]$-uniform $\mathsf{SIZE\text{-}DEPTH}[n \cdot (\log n)^c, (\log n)^d]$. We will show that this assumption contradicts the space hierarchy theorem.

By Lemma 18, we conclude for constructible $k(n) = o(\log n/\log\log n)$ that

$$\Sigma_{k(n)}\mathsf{TIME}[n] \subseteq \mathsf{SPACE}[(\log n)^d]\text{-uniform } \mathsf{SIZE\text{-}DEPTH}[n \cdot (\log n)^{(c+1)\cdot k(n)}, (\log n)^d]. \quad (2)$$

Set $s(n) = (\log n)^{d'}$ in the following, where $d < d' < 2$. Applying Lemma 17 with $a(n) = 0$, $t_1(n) = O(n)$, $t_2(n) = s(n)/\log n$, and $s(n)$ arbitrary, we have the containment

$$\mathsf{SPACE}[s(n)] = \mathsf{TISP}[2^{O(s(n))}, s(n)] \subseteq \Sigma_{s(n)/\log n}\mathsf{TIME}[n \cdot s(n)^2/\log n].$$

Setting $k(n) = s(n)/\log n$ and noting that $s(n)/\log n = (\log n)^{d'-1} = o(\log n/\log\log n)$, we derive from the containment (2) that

$$\Sigma_{s(n)/\log n}\mathsf{TIME}[n \cdot s(n)^2/\log n]$$

is contained in

$$\begin{aligned}
&\mathsf{SPACE}[(\log n)^d]\text{-uniform } \mathsf{SIZE\text{-}DEPTH}[n \cdot s(n)^2 \cdot (\log n)^{(c+1)\cdot s(n)/\log n}, O((\log n)^d)] \\
&\subseteq \mathsf{SPACE}[(\log n)^d]\text{-uniform } \mathsf{DEPTH}[O((\log n)^d)] \\
&\subseteq \mathsf{SPACE}[o(s(n))]\text{-uniform } \mathsf{DEPTH}[o(s(n))] \\
&\subseteq \mathsf{SPACE}[o(s(n))]. \qquad \text{(Proposition 12)}
\end{aligned}$$

We have derived $\mathsf{SPACE}[s(n)] \subseteq \mathsf{SPACE}[o(s(n))]$, which contradicts the space hierarchy theorem [27]. ◀

5 Conclusion

In this paper, we showed how to "amplify" small-circuit upper bounds in a new way: if a simple circuit composition problem has nearly-linear size circuits, then a much larger class of problems also has nearly-linear size circuits. This led to new circuit lower bounds and connections between lower bound problems. Many open problems have naturally arisen.

- We have shown that $\mathsf{TIME}[n^{1+\varepsilon}]$ does not have $\mathsf{LOGSPACE}$-uniform linear-size circuits, and the lower bound is non-relativizing. Can this be strengthened to P-uniform linear circuits? Alternatively, can our lower bounds for circuit composition be generalized to prove that $\mathsf{TIME}[n^k] \not\subseteq \mathsf{LOGSPACE}$-uniform $\mathsf{SIZE}[n^{k-\varepsilon}]$, for any constant k?
  We conjecture that the circuit composition problems defined in this paper, especially General Circuit n-Composition, require *non-uniform* super-linear-size circuits. The fact that we can at least rule out $\mathsf{LOGSPACE}$-uniform circuits gives some hope that future work can relax the uniformity conditions.
- What additional consequences can be derived from assuming $\mathsf{NP} \subset \mathsf{SIZE}[O(n)]$? How well can PSPACE-complete problems like QBF be solved with circuits, under this assumption? From the results of this paper, we have that QBF has $2^{\tilde{O}(\sqrt{n})}$-size circuits, assuming SAT is in $\mathsf{SIZE}[O(n)]$ or assuming $\mathsf{TIME}[n^{1+\varepsilon}]$ is in $\mathsf{SIZE}[O(n)]$. Is it possible that $\mathsf{NP} \subset \mathsf{SIZE}[O(n)]$ implies $\mathsf{PSPACE} \subset \mathsf{P/poly}$?
- Can we prove $\mathsf{P} \not\subset \mathsf{P}^{\mathsf{NP}}$-uniform $\mathsf{SIZE}(O(n))$? Is the problem equivalent to $\mathsf{P} \not\subset \mathsf{SIZE}(O(n))$? A yes-answer would show that constructing these linear-size circuits cannot even be done by a P^{NP} process, progressing even closer to $\mathsf{P} \not\subset \mathsf{SIZE}(O(n))$. We observe that $\mathsf{P} \not\subset \mathsf{SIZE}(O(n))$ is in fact equivalent to $\mathsf{P} \not\subset \mathsf{P}^{\Sigma_2\mathsf{P}}$-uniform $\mathsf{SIZE}(O(n))$: in $\mathsf{P}^{\Sigma_2\mathsf{P}}$ one can guess and verify a linear-size circuit for a polynomial-time computation.

Acknowledgments. We thank the CCC referees for their helpful comments.

References

1 Miklos Ajtai. Determinism versus non-determinism for linear time RAMs (extended abstract). In *STOC: ACM Symposium on Theory of Computing (STOC)*, 1999.

2 Eric Allender and Michal Koucký. Amplifying lower bounds by means of self-reducibility. *J. ACM*, 57(3):14:1–14:36, 2010.

3 Eric Allender, Michal Koucký, Detlef Ronneburger, Sambuddha Roy, and V. Vinay. Time-space tradeoffs in the counting hierarchy. In *Proceedings of the 16th Annual IEEE Conference on Computational Complexity, Chicago, Illinois, USA, June 18-21, 2001*, pages 295–302, 2001.

4 Sanjeev Arora and Boaz Barak. *Complexity Theory: A Modern Approach*. Cambridge University Press, Cambridge, 2009.

5 Paul Beame, Michael E. Saks, Xiaodong Sun, and Erik Vee. Time-space trade-off lower bounds for randomized computation of decision problems. *J. ACM*, 50(2):154–195, 2003.

6 Samuel R. Buss and Ryan Williams. Limits on alternation trading proofs for time-space lower bounds. *Computational Complexity*, 24(3):533–600, 2015.

7 Stephen A. Cook. Deterministic CFL's are accepted simultaneously in polynomial time and log squared space. In *STOC*, pages 338–345, 1979.

8 Ning Ding. Some new consequences of the hypothesis that P has fixed polynomial-size circuits. In *Theory and Applications of Models of Computation TAMC*, pages 75–86, 2015.

9 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. *Electronic Colloquium on Computational Complexity (ECCC)*, 22:166, 2015. URL: `http://eccc.hpi-web.de/report/2015/166`.

10 Lance Fortnow. Time-space tradeoffs for satisfiability. *Journal of Computer and System Sciences*, 60(2):337–353, April 2000.

11 Lance Fortnow, Richard Lipton, Dieter van Melkebeek, and Anastasios Viglas. Time-space lower bounds for satisfiability. *J. ACM*, 52(6):833–865, 2005.

12 Lance Fortnow, Rahul Santhanam, and Ryan Williams. Fixed polynomial size circuit bounds. In *Proceedings of 24th Annual IEEE Conference on Computational Complexity*, pages 19–26, 2009.

13 Juris Hartmanis and Richard Stearns. On the computational complexity of algorithms. *Trans. Amer. Math. Soc. (AMS)*, 117:285–306, 1965.

14 Frederick Hennie and Richard Stearns. Two-tape simulation of multitape Turing machines. *Journal of the ACM*, 13(4):533–546, October 1966.

15 John Hopcroft, Wolfgang Paul, and Leslie Valiant. On time versus space. *Journal of the ACM*, 24(2):332–337, April 1977.

16 Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *Journal of Computer and System Sciences*, 62(4):512–530, 2001.

17 Richard Lipton. Some consequences of our failure to prove non-linear lower bounds on explicit functions. In *Proceedings of 9th Annual Structure in Complexity Theory Conference*, pages 79–87, 1994.

18 Richard J. Lipton and Anastasios Viglas. Non-uniform depth of polynomial time and space simulations. In *Proceedings of Fundamentals of Computation Theory, 14th International Symposium (FCT)*, pages 311–320, 2003.

19 Richard J. Lipton and Ryan Williams. Amplifying circuit lower bounds against polynomial time, with applications. *Computational Complexity*, 22(2):311–343, 2013.

20 V. Nepomnjascii. Rudimentary predicates and turing calculations. *Soviet Mathematics – Doklady*, 11(6):1462–1465, 1970.

21 Nicholas Pippenger. On simultaneous resource bounds (preliminary version). In *FOCS*, pages 307–311, 1979.

22 Nicholas Pippenger and Michael J. Fischer. Relations among complexity measures. *J. ACM*, 26(2):361–381, 1979.

23 Rüdiger Reischuk. *Einführung in die Komplexitätstheorie*. Teubner, 1990.

24 Walter L. Ruzzo, Janos Simon, and Martin Tompa. Space-bounded hierarchies and probabilistic computations. *J. Comput. Syst. Sci.*, 28(2):216–230, 1984.

25 Rahul Santhanam and Ryan Williams. On uniformity and circuit lower bounds. *Computational Complexity*, 23(2):177–205, 2014. Preliminary version in CCC'13.

26 Richard Stearns, Juris Hartmanis, and Philip Lewis. Hierarchies of memory limited computations. In *Proceedings of the Sixth Annual Symposium on Switching Circuit Theory and Logical Design*, pages 179–190. IEEE, 1965.

27 Richard Stearns, Juris Hartmanis, and Philip Lewis. Hierarchies of memory limited computations. In *Proceedings of the Sixth Annual Symposium on Switching Circuit Theory and Logical Design*, pages 179–190. IEEE, 1965.

28 Dieter Van Melkebeek. *A survey of lower bounds for satisfiability and related problems*, volume 7. Now Publishers Inc, 2007.

29 Heribert Vollmer. *Introduction to circuit complexity: a uniform approach*. Springer Science & Business Media, 1999.

30 R. Ryan Williams. Time-space tradeoffs for counting NP solutions modulo integers. *Computational Complexity*, 17(2):179–219, 2008.

31 Ryan Williams. Alternation-trading proofs, linear programming, and lower bounds. *TOCT*, 5(2):6, 2013.

32 Christopher Wilson. Relativized circuit complexity. *Journal of Computer and System Sciences*, 31(2):169–181, 1985.

33 Stanislav Žák. A Turing machine time hierarchy. *Theoretical Computer Science*, 26(3):327–333, October 1983.

A Oracle Relative to Which Polytime has Small Constructible Circuits

In the following, we use the standard model of oracle computation for logarithmic space of Ruzzo, Simon, and Tompa [24]: there is an special oracle tape that is write-only, its content does not count towards the space bound, and the oracle tape is erased after each oracle query.

▶ **Theorem 19.** *For every $k \geq 1$, there is an oracle B such that every language solvable in time n^k with an oracle for B has $O(n)$-size B-oracle circuits constructible in logspace equipped with an oracle for B.*

Proof. Our construction is very similar to that of Wilson [32], who shows there are oracles relative to which P has size $O(n)$ circuits. To make the circuits logspace-constructible as well, we add a simple but crucial modification to the oracle, which relies on having a fixed polynomial upper bound on the running time.

Fix a constant $k \geq 1$. In the following, let $\{M_i\}$ be an enumeration of machines running in at most $n^k + k$ steps, and let $\langle \cdot, \cdot \rangle$ be an efficient pairing function. We construct the oracle B in stages. In stage 0, we assign B to be the empty set.

In stage n:

- Start with an empty set S_n. For every integer $1 \leq i \leq n$ and n-bit string x, execute M_i on x with the current oracle B for $n^k + k$ steps. If M_i accepts x, put the pair $\langle i, x \rangle$ in S_n.
- "Mark" every string that is queried by the M_i's among these 2^n executions. Note there are at most $(n^k + k)n2^n$ strings marked in this step, and the total number of marked strings (over all stages $0, \ldots, n$) at this point is at most $(n^k + k)n2^{n+1}$.
- Let y_n be a string of length $n + (k+1)\log n + 3k$ such that $\{\langle i, x \rangle y_n \mid \langle i, x \rangle \in S_n\}$ contains no marked strings. There are $t = n^{k+1}2^{n+3k}$ such strings y_n, so we are considering $t > (n^k + k)n2^{n+1}$ different sets of strings over $\{0,1\}^{2n+(k+1)\log n+3k}$. Hence at least one of the sets does not contain a marked string. Put all $\langle i, x \rangle y_n$ in the oracle B.

- Finally, put the set of strings $\{0^{n^{k+1}+k+1}1j \mid \text{the } j\text{th bit of } y_n \text{ is } 1\}$ in B as well, where the j's are construed as $\lceil \log_2(n + k \log n + 3k + 1) \rceil$-bit strings. Note that none of the strings in this set can be marked in stage n, because all of them have length greater than $n^{k+1} + (k+1)$, and no M_i can query a string longer than $n^k + k$ over inputs x of length n.

Now, for any machine M_i and sufficiently large $n \gg i$, our circuit C_n computing M_i on all inputs of length n consists of the single gate

$$B(\langle i, x \rangle y_n),$$

with the index i and the string y_n hard-coded in C_n. By our construction of B, there is a chosen string y_n of length $O(n)$ for which deciding $\langle i, x \rangle y_n \in B$ is equivalent to deciding if M_i accepts x. The total number of wires (and hence size) in the circuit C_n is $O(n)$.

Furthermore, for every machine M_i running in n^k time, the circuits C_n for M_i can also be constructed in $O(k \log n)$ space with an oracle for B. Given the string 1^n, our logspace machine L_i for printing C_n first prints the index i of M_i in some natural encoding, and prints n "sources" of C_n, indicating the inputs $x_1, \ldots, x_n$. To construct the string y_n, L_i uses the oracle B. In particular, the logspace machine L_i has a counter which holds some integer $j = 1, \ldots, n + k \log n + 3k$. For each j, L_i prints $0^{n^{k+1}+k+1}1j$ on the write-only oracle tape (by maintaining a counter $\ell = 1, \ldots, n^{k+1} + k + 1$ for printing the zeroes) then queries B. The j-th query answer tells L_i the j-th bit of y_n, so L_i can output these bits as part of the description of C_n. ◀

This oracle is especially interesting when contrasted with the lower bound of Santhanam and Williams [25] that $\mathsf{P} \not\subset \mathsf{P}$-uniform $\mathsf{SIZE}(O(n))$. The proof of that lower bound *does* relativize (and thus is true for all oracles). The key difference between $\mathsf{P} \not\subset \mathsf{P}$-uniform $\mathsf{SIZE}(O(n))$ and the (non-relativizing) lower bound of Theorem 2 (general circuit composition is not in LOGSPACE-uniform $n^{1+o(1)}$ size) seems to be that in the former, there is no fixed polynomial upper bound on the complexity class (P) that is being simulated.

B Subexponential-Size Circuits for QBF from Small-Size Circuits for SAT

We can also derive interesting new consequences from the assumption that the SAT problem has $\tilde{O}(n)$-size circuits. They follow without much difficulty from the literature on SAT time-space tradeoffs, but we feel the connections are worth recording.

▶ **Claim 20** ([10]). *If SAT* $\in \mathsf{TIME}[O(n)]$ *then there is a* $c > 0$ *such that for all* k, $\Sigma_k\mathsf{TIME}[O(n)] \subseteq \mathsf{TIME}[n \cdot (\log n)^{ck}]$.

Proof. It is enough to show that SAT $\in \mathsf{TIME}[O(n)]$ implies that

$$\Sigma_k\mathsf{TIME}[O(n)] \subseteq \Sigma_{k-1}\mathsf{TIME}[n \cdot (\log n)^d]$$

for a fixed constant $d > 0$. Suppose SAT $\in \mathsf{TIME}[O(n)]$. Then by an efficient Cook-Levin Theorem (see for example [11]), we have $\mathsf{NTIME}[O(n)] \subseteq \mathsf{TIME}[\tilde{O}(n)]$ and $\mathsf{coNTIME}[O(n)] \subseteq \mathsf{TIME}[\tilde{O}(n)]$. Let $L \in \Sigma_k\mathsf{TIME}[O(n)]$ have an acceptance condition of the form:

$$\exists x_1 \forall x_2 \ldots Q_k x_k M(x, x_1, x_2, \ldots, x_k) \tag{3}$$

where M is a deterministic $O(n)$-time machine, and all x_i's are $O(n)$-length strings. Then on the tuple of strings $(x, x_1, x_2, \ldots, x_{k-1})$, the expression

$$Q_k x_k M(x, x_1, x_2, \ldots, x_k) \tag{4}$$

represents a computation in $\mathsf{coNTIME}[O(n)]$, which by hypothesis is computable in $\tilde{O}(n)$ time. Let N be a deterministic machine that computes the value of (4) in $\tilde{O}(n)$ time, given $(x, x_1, x_2, \ldots, x_{k-1})$ as input. Then deciding the truth of

$$\exists x_1 \forall x_2 \ldots Q_{k-1} x_{k-1} N(x, x_1, \ldots, x_{k-1})$$

is equivalent to deciding (3). Thus L can be decided in $\Sigma_{k-1}\mathsf{TIME}[\tilde{O}(n)]$, which completes the proof. ◀

▶ **Corollary 21.** *If* $SAT \in \mathsf{TIME}[O(n)]$ *then* $\mathsf{LOGSPACE} \subseteq \bigcup_k \mathsf{TIME}[n \cdot (\log n)^k]$.

Proof. We know that for every $L \in \mathsf{LOGSPACE}$ there is a k such that $L \in \Sigma_k\mathsf{TIME}[O(n)]$. Apply the above claim. ◀

▶ **Claim 22.** *If* $SAT \in \mathsf{SIZE}[O(n)]$ *then there is a* c *such that for all* k, $\Sigma_k\mathsf{TIME}[O(n)] \in \mathsf{SIZE}[n(\log n)^{ck}]$.

Proof. Similar to Claim 20. If $\mathrm{SAT} \in \mathsf{SIZE}[O(n)]$ then $\mathsf{NTIME}[O(n)] \subseteq \mathsf{SIZE}[\tilde{O}(n)]$, which means that

$$\Sigma_k\mathsf{TIME}[O(n)] \subseteq \Sigma_{k-1}\mathsf{TIME}[n \log^c n]/(n \log^c n) \subseteq \ldots$$

$$\subseteq \mathsf{TIME}[n \log^{ck} n]/(n \log^{ck} n) \subseteq \mathsf{SIZE}[n(\log n)^{ck}]\,. \quad ◀$$

▶ **Corollary 23.** *If* $SAT \in \mathsf{SIZE}[O(n)]$ *then* $\mathsf{LOGSPACE} \subseteq \bigcup_k \mathsf{SIZE}[n \cdot (\log n)^k]$.

Specifically, we can relate the circuit complexity of SAT and QBF as follows:

▶ **Reminder of Theorem 7.** *If* SAT *is in* $\mathsf{SIZE}[\tilde{O}(n)]$ *then* QBF *is in* $\mathsf{SIZE}[2^{O(\sqrt{n \log n})}]$.

Proof. Suppose $\mathrm{SAT} \in \mathsf{SIZE}[\tilde{O}(n)]$. By Claim 22, we have

$$\Sigma_{k(n)}\mathsf{TIME}[O(n)] \subseteq \mathsf{SIZE}[n(\log n)^{c \cdot k(n)}]. \tag{5}$$

Let $a(n) = 0$, $s(n) = O((\log n)^2/\log\log n)$, $t_2(n) = k(n)$, and $t_1(n) = n$. Applying Lemma 17, we have

$$\mathsf{TISP}[n^{k(n)}, O((\log n)^2/\log\log n)] \subseteq \Sigma_{k(n)}\mathsf{TIME}[\tilde{O}(n)]. \tag{6}$$

Setting $k(n) = \varepsilon \log n/(\log\log n)$ for sufficiently small $\varepsilon > 0$, we have

$$\mathsf{SPACE}[\varepsilon(\log n)^2/\log\log n] \subseteq \mathsf{TISP}[n \cdot 2^{\varepsilon(\log n)^2/\log\log n}, \varepsilon(\log n)^2/\log\log n]$$

$$\subseteq \mathsf{TISP}[n^{1+\varepsilon \log n/\log\log n}, \varepsilon(\log n)^2/\log\log n] \subseteq \Sigma_{1+\varepsilon \log n/\log\log n)}\mathsf{TIME}[\tilde{O}(n)]$$

$$\subset \mathsf{SIZE}[n(\log n)^{1+\varepsilon \log n/\log\log n)}] = \mathsf{SIZE}[n^{1+2\varepsilon}].$$

By padding each language in $\mathsf{SPACE}[\varepsilon(\log n)^2/\log\log n]$ to size $N = 2^{O(\sqrt{n \log n})}$, we have $(\log N)^2/\log\log N \leq O(n \log n/\log n) \leq O(n)$ and $N^2 \leq 2^{O(\sqrt{n \log n})}$, we can conclude that $\mathsf{SPACE}[O(n)]$, which includes QBF, has circuits of size $2^{O(\sqrt{n \log n})}$. ◀

PPSZ for General k-SAT – Making Hertli's Analysis Simpler and 3-SAT Faster*

Dominik Scheder[1] and John P. Steinberger[2]

1 **Department of Computer Science and Engineering, Shanghai Jiaotong University, Shanghai, China**
dominik@cs.sjtu.edu.cn

2 **Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing, China**
jpsteinb@gmail.com

Abstract

The currently fastest known algorithm for k-SAT is PPSZ named after its inventors Paturi, Pudlák, Saks, and Zane [7]. Analyzing its running time is much easier for input formulas with a unique satisfying assignment.

In this paper, we achieve three goals. First, we simplify Hertli's 2011 analysis [1] for input formulas with multiple satisfying assignments. Second, we show a "translation result": if you improve PPSZ for k-CNF formulas with a unique satisfying assignment, you will immediately get a (weaker) improvement for general k-CNF formulas.

Combining this with a result by Hertli from 2014 [2], in which he gives an algorithm for Unique-3-SAT slightly beating PPSZ, we obtain an algorithm beating PPSZ for general 3-SAT, thus obtaining the so far best known worst-case bounds for 3-SAT.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Boolean satisfiability, exponential algorithms, randomized algorithms

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.9

1 Introduction

The problem of SAT, deciding whether a proposition formula conjunctive normal form has a satisfying assignment (or even constructing such a solution) enjoys a central position among NP-complete problems. The case of k-SAT, in which the input is restricted to k-CNF formulas, i.e., formulas of clause width bounded by k, has drawn special attention. An obvious brute-force algorithm solves SAT in time $O(2^n \mathrm{poly}(n))$, where n is the number of variables. For k-SAT, this running time has been improved quite a bit. Two approaches stand out: local search algorithms and encoding based algorithms. In 1999, Schöning [11] gave a simple local search algorithm for k-SAT. Paturi, Pudlák, and Zane [8] came up with an encoding-based algorithm, called PPZ in their honor. PPZ is not as good as Schöning, but has interesting applications in circuit complexity [8] and complexity of exponential algorithms [4].

Most importantly for this paper, there exists a "PPZ 2.0 version" called PPSZ (Paturi, Pudlák, Saks, and Zane [7]). This is the currently fastest randomized algorithm for k-SAT.

* Dominik Scheder gratefully acknowledges support by the National Natural Science Foundation of China under grant 61502300.

32nd Computational Complexity Conference (CCC 2017).
Editor: Ryan O'Donnell; Article No. 9; pp. 9:1–9:15

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

It is quite simple to state but challenging to analyze. We should state that its actual worst-case running time is not understood at all: Chen, Scheder, Talebanfard, Tang [10] construct exponentially hard instances, but their bounds are quite poor. Perhaps counterintuitively, the analysis in [7] incurs an exponential loss if the input formula has multiple solutions. Only in 2011, Timon Hertli [1] closed this gap in a breakthrough paper by a better (and simpler, yet still quite challenging) analysis. Still, PPSZ continued to be the best algorithm. A first crack in the wall appeared in 2014, when Hertli [2] combined PPSZ with several other algorithms, and showed that this improves the running time of Unique-3-SAT by a small but exponential amount. By *Unique-k-SAT* we mean k-SAT where the input formula F can have at most one satisfying assignment. If F may have multiple solutions, we write *general* k-SAT.

In this paper we first give a simpler analysis of Hertli's 2011 result [1]. This analysis also yields a translation result: if you improve PPSZ for Unique-k-SAT, you immediately get a (smaller) improvement for general k-SAT. Thus, researchers who want to "crack the PPSZ barrier" can focus on Unique-k-SAT for the time being. This, together with Hertli's 2014 improvement for Unique-3-SAT [2], gives the currently fastest known running time for general 3-SAT.

To give the reader an impression of which running time we are talking about, let us state some bounds for 3-SAT, ignoring subexponential factors. PPZ [8] runs in time $O\left(2^{2n/3}\right) \approx O\left(1.59^n\right)$, Schöning [11] in time $O\left(\left(\frac{4}{3}\right)^n\right) \approx O\left(1.334^n\right)$, and PPSZ [7] in time $O\left(2^{(2\ln(2)-1)n}\right) \approx O\left(1.308^n\right)$. The improvements by Hertli [2] and this paper are quite small (think of in the ballpark of tenth digit after the dot) and serve more as a demonstration that PPSZ *can* be improved, even if they do not improve it by much.

1.1 The PPSZ Algorithm

PPSZ is a probabilistic algorithm that tries to incrementally construct a satisfying assignment of F. The "generic PPSZ algorithm" is easy to state. Given a k-CNF formula F, choose a variable x therein uniformly at random; then choose a value $b \in \{0,1\}$. Choose b uniformly at random, unless we can determine the "correct" truth value of x by some correct yet incomplete proof heuristic.

Let us state things more formally. A proof heuristic is a deterministic procedure P which on input F and x outputs a value $b \in \{0,1,?\}$. Correctness means that $P(F,x) = b \in \{0,1\}$ means that $F \models (x = b)$, i.e., b is really the correct value of x; incompleteness means that we allow $P(F,x)$ to output "?", even if only one value $b \in \{0,1\}$ for x is feasible. From now on, when we say *proof heuristic*, we always mean a correct but possibly incomplete heuristic.

Suppose now that $\alpha \in \mathrm{sat}(F)$, i.e., it is a satisfying assignment. Below we give procedure Encode that, given access to α, F, the heuristic P, and a permutation π of the variables of F, encodes α into a bit string c, hopefully using fewer than n bits. Intuitively, it iterates through the variables in the order given by π and outputs $\alpha(x)$ for every variable, unless this value is already implied by F and the bits output so far. This encoding is reversible: the procedure Decode can recover α when given access to F, P, π, and the encoding c. The generic algorithm RandomDecode then is simply to choose π and c randomly, start decoding and hoping for the best.

Note that the running time of RandomDecode is dominated by the running time of P. Thus, as long as P runs in polynomial (subexponential) time, so does RandomDecode. Consequently, we measure the goodness of RandomDecode not in terms of running time, but in terms of *success probability*, which will usually be of the form 2^{-pn} for some constant p. To make RandomDecode into an algorithm, we still have to specify P. Here are some examples:

Algorithm 1 Generic Encoding Procedure

```
1: procedure Encode(α, π, F, P)
2:     β := the empty assignment on V
3:     for x ∈ V in the order of π do
4:         if P(F|_β, x) =? then
5:             output α(x)
6:         end if
7:         add [x ↦ α(x)] to β
8:     end for
9: end procedure
```

Algorithm 2 Generic Decoding Procedure

```
1: procedure Decode(c, π, F, P)
2:     β := the empty assignment on V
3:     for x ∈ V in the order of π do
4:         if P(F|_β, x) = b ∈ {0,1} then
5:             β(x) := b
6:         else
7:             β(x) := the next bit of c
8:         end if
9:     end for
10:    return β
11: end procedure
```

Algorithm 3 Generic Random Decoding Procedure

```
1: procedure RandomDecode(F, P)
2:     π := a random permutation on V
3:     c := a random string in {0,1}^n
4:     β := Decode(c, π, F, P)
5:     return β if it satisfies F, else failure
6: end procedure
```

Example: P_0. This heuristic always outputs "?". Obviously, RandomDecode(F, P_0) is just random guessing, and each solution α appears with probability 2^{-n}. This is not a very good algorithm.

Example: P_1. This heuristic answers $P_1(F, x) = b \in \{0, 1\}$ if F is a CNF formula and F contains the unit clause $(x = b)$[1] RandomDecode(F, P_1) is the algorithm PPZ, invented by Paturi, Pudlák, and Zane [7]. Its success probability on k-CNF formulas is $2^{-(1-1/k)n}$.

Example: P_d. This heuristic generalizes P_1. It answers $P_d(F, x) = b$ if F is a CNF formula and it contains a subset G of at most d clauses for which $G \models (x = b)$. With this heuristic, RandomDecode(F, P_d) becomes PPSZ, although Paturi, Pudlák, Saks, and Zane[7] state

[1] If F contains both $(x = 0)$ and $(x = 1)$ then $P_1(F, x)$ can be either 0 or 1, but in this case F is unsatisfiable anyway.

it slightly differently. Its success probability is much higher than that of PPZ (we will give more details below) but it is still not completely understood.

Example: P_∞. This heuristic employs the whole power of propositional logic. It answers $P_\infty(F, x) = b \in \{0, 1\}$ if F implies $(x = b)$. Obviously, determining this is itself NP-hard, so this is not an efficient heuristic. Still, it will be important in this paper. Note that for satisfiable F, RandomDecode(F, P_∞) always outputs a solution. Thus, it defines a distribution Q on pairs (π, α), where π is the permutation is chooses and α the solution it outputs. The distribution Q will be very important in our proofs below.

1.2 Gauging the Strength of the Proof Heuristic P

Towards an analysis of its success probability time, let $C_x(\alpha, \pi)$ be the indicator variable which is 1 if Encode outputs a bit for x., i.e., if $P(F|_\beta, x) = ?$ in Line 4 of Encode. So $C(\pi, \alpha) := \sum_x C_x(\pi, \alpha)$ is the length of the encoding, i.e., $|c| = C(\pi, \alpha)$. Note that $C_x(\pi, \alpha)$ also depends on F and P. Since they are usually fixed throughout, we choose to drop them for the sake of readability.

▶ **Observation 1.** $\Pr[\text{RandomDecode}(F, P) \text{ returns } \alpha] = \mathbb{E}_\pi \left[2^{-C(\pi,\alpha)}\right]$.

Proof. Let $c^* := \text{Encode}(\alpha, \pi, F, P)$. RandomDecode returns α iff the first $C(\pi, \alpha)$ bits of its random string $c \in \{0, 1\}^n$ agree with c^*. ◀

We write $F \models T$ as a shorthand of "F implies T", i.e., every satisfying assignment of F satisfies T. If $F \models (x = 0)$ or $F \models (x = 1)$ we say that x is *frozen in* F. Equivalently, all satisfying assignments of F agree on x. Otherwise, we say that x is *liquid*. Note that $C_x(\pi, \alpha)$ can be 1 for two reasons. First, it could be that in Line 4 of Encode, x is liquid in $F|_\beta$ and thus every correct proof heuristic P must answer $P(F|_\beta, x) = ?$. In this case we set $I_x(\pi, \alpha) = 1$. Second, it could be that x is frozen in $F|_\beta$ and therefore $P(F|_\beta, x) = ?$ is due to the incompleteness of P. In this case we set $J_x(\pi, \alpha) = 1$. Thus, $C_x(\pi, \alpha) = I_x(\pi, \alpha) + J_x(\pi, \alpha)$. We also set $I(\pi, \alpha) = \sum_x I_x(\pi, \alpha)$ and $J(\pi, \alpha) = \sum_x J(\pi, \alpha)_x$. Note that $I(\pi, \alpha) = 0$ if F has a unique satisfying assignment, since all variables are frozen. Also, $J(\pi, \alpha) = 0$ for P_∞, since this heuristic never fails. Here is a plausible notion of strength for proof heuristics: if P is a strong proof heuristic, then $J_x(\pi, \alpha) = 1$ should not happen too often:

▶ **Definition 2** (Error of P)**.** Let $\mathcal{C}$ be a class of formulas and P be a proof heuristic. P has *error at most p against $\mathcal{C}$* if $\mathbb{E}_\pi [J_x(\pi, \alpha)] \le p$ for every $F \in \mathcal{C}$, solution α, and variable x in F.

▶ **Theorem 3** ([8])**.** *P_1 has error $1 - 1/k$ against k-CNF formulas.*

Paturi, Pudlák, Saks, and Zane[7] prove the following bound on the error of P_d (although they do not use this exact wording). Consider the infinite $(k - 1)$-ary rooted tree. For each vertex v in this tree, choose $\pi_v \in [0, 1]$ uniformly at random. Delete each vertex v with $\pi_v < \pi_{\text{root}}$. Let s_k be probability that the root is contained in an infinite connected component. It is easy to see that $s_2 = 0$. A simple calculation shows that $s_3 = 2\ln(2) - 1$.

▶ **Theorem 4** ([7])**.** *P_d has error $s_k + \epsilon_{d,k}$ against k-CNF formulas, where $\epsilon_{d,k} \to 0$ as $d \to \infty$.*

▶ **Observation 5.** *Let P be a proof heuristic of error at most p against $\mathcal{C}$. If $F \in \mathcal{C}$ has a unique satisfying assignment α, then* RandomDecode$(F, P) = \alpha$ *with probability at least* 2^{-pn}.

Proof. We use Observation 1 and Jensen's Inequality:

$$\begin{aligned}
\Pr[\text{ PPSZ succeeds}] &= \mathop{\mathbb{E}}_{\pi}\left[2^{-C(\pi,\alpha)}\right] \geq 2^{-\mathbb{E}_\pi[C(\pi,\alpha)]} && \text{(Jensen's Inequality)}\\
&= 2^{-\mathbb{E}_\pi[J(\pi,\alpha)]} && (I = 0 \text{ since only one assignment})\\
&\geq 2^{-pn} && (P \text{ has error at most } p)
\end{aligned}$$

◀

1.3 Previous Work

In case F has multiple satisfying assignments, the proof of Observation 5 breaks down, and it is not clear why a proof heuristic of error at most p should give an algorithm of success probability 2^{-pn}. A series of authors have improved PPSZ for the general case of multiple satisfying assignments. Paturi, Pudlák, Saks, and Zane [7] already gave an analysis, which has an exponential loss for $k = 3, 4$. Iwama and Tamaki [6] combine PPSZ for Schöning's random walk algorithm [11] to obtain a better algorithm. This combination was then further explored by Rolf [9], Iwama, Seto, Takai, and Tamaki [5], and Hertli, Moser, and Scheder [3]. All these improvements have serious drawbacks: they still have an exponential loss compared to the Unique-k-SAT bound for $k = 3, 4$; they are extremely technical; they use detailed knowledge of the proof heuristic P; finally, the latter four have to combine PPSZ with a second algorithm (Schöning's random walk algorithm [11]) to achieve their improvement. In 2011, Timon Hertli achieved a breakthrough by proving the following theorem:

▶ **Theorem 6** (Hertli [1])**.** *Suppose P has error at most p against $\mathcal{C}$, and $p \geq p^* := \frac{2-\log(e)}{2} \approx 0.279$. For every satisfiable $F \in \mathcal{C}$,* RandomDecode(F, P) *returns a satisfying assignment with probability at least* 2^{-pn}.

Note the mysterious p^* in the theorem. We suspect that it is an artefact of the proof and make the following conjecture:

▶ **Conjecture 7.** *Theorem 6 holds for all $p \geq 0$.*

Currently, the only supporting evidence for the conjecture is (1) our failure to construct a counterexample, despite some trying, and (2) that it would simply be very weird if it were false. Anyway, since $1 - s_k \geq p^*$ for all $k \geq 3$, Hertli's theorem works for the current version of PPSZ, for all $k \geq 3$. It might be, however, that future research brings about proof heuristics of error probability less than p^*, in which case the above theorem would again incur an exponential loss. Ingenious as it is, Hertli's proof is quite long and tedious.

1.4 Our Contribution

The first contribution of this paper is to give a much simpler proof of Theorem 6. Our proof in fact highlights why certain previous attempts fail, demonstrates more clearly "what is going on", and also points towards further improvements.

As a second contribution, we show that any improvement of PPSZ for Unique-k-SAT translates into a (weaker) improvement for General k-SAT. In particular, we will prove a stronger version of Theorem 6, which we now explain.

▶ **Definition 8.** A class $\mathcal{C}$ of formulas or circuits is *closed under restrictions* if $F \in \mathcal{C}$ implies that $F|_{x=b} \in \mathcal{C}$, for every variable x and value $b \in \{0,1\}$.

Note that this applies to most "reasonable" circuit classes, in particular to k-CNF formulas.

▶ **Definition 9.** A proof heuristic P is called *monotone* if $P(F,x) \in \{0,1\}$ implies that $P(F|_{y=b},x) \in \{0,1\}$, for every F, $y \neq x$, and $b \in \{0,1\}$.

In other words, if P can deduce the value of x, then it can also do so after we add the additional information that $y = b$. Note that P_0, P_1, P_d, P_∞ define above are all monotone. Recall that RandomDecode(F, P_∞) chooses a uniformly random permutation $\pi \in \mathrm{Sym}(V)$ and always outputs a satisfying assignment. Thus, it defines a distribution Q on $\mathrm{Sym}(V) \times \mathrm{sat}(F)$ with $Q(\pi,\alpha) = \frac{1}{n!} \cdot 2^{-I(\pi,\alpha)}$.

▶ **Theorem 10.** *Suppose P has error at most p against $\mathcal{C}$, and set $q := p - p^*$ for $p^* := \frac{2-\log(e)}{2} \approx 0.279$. Let $F \in \mathcal{C}$ be satisfiable. Then* RandomDecode *returns a satisfying assignment with probability at least $2^{-pn+q\,\mathbb{E}_{(\pi,\alpha)\sim Q}[I(\pi,\alpha)]}$, where $q := p - p^*$.*

Since $s_k > p^*$ for all $k \geq 3$, the value q above is positive, which immediately reproves Hertli's Theorem (Theorem 6). As pointed out by one of the referees, the "bonus term" $\mathbb{E}_{(\pi,\alpha)\sim Q}[I(\pi,\alpha)]$ has an information-theoretic interpretation: it is the conditional entropy $H(\alpha|\pi)$. Our theorem has a nice by-product, a "translation result" from Unique-k-SAT to General k-SAT: suppose you have an algorithm A which is exponentially better than PPSZ for Unique-k-SAT. Given an input k-CNF formula F, there are two cases: first, it could be that $\mathbb{E}_Q[I]$ is "large" for this F, in which case Theorem 10 already gives an exponential bonus; or it is "small", in which case there is a small restriction ρ such that $F|_\rho$ has a unique satisfying assignment. We can now guess ρ and apply A to $F|_\rho$. Formally, we obtain the following theorem:

▶ **Theorem 11.** *Suppose P is a monotone proof heuristic with error probability at most p against class $\mathcal{C}$. We assume that $\mathcal{C}$ is closed under restrictions.*

1. *If* RandomDecode$(P,\cdot)$ *solves* Unique-$\mathcal{C}$-SAT *with probability at least $2^{(-p+\epsilon)n}$, then it solves $\mathcal{C}$-SAT with probability at least $2^{(-p+\epsilon')n}$.*
2. *If there is an algorithm A for* Unique-$\mathcal{C}$-SAT *with success probability $2^{(-p+\epsilon)n}$, then there is an algorithm A' for $\mathcal{C}$-SAT with success probability at least $2^{(-p+\epsilon')n}$ and running time n times that of A.*
3. *If there is Monte Carlo algorithm B solving* Unique-$\mathcal{C}$-SAT *running in time $2^{(p-\epsilon)n}$, then there exists a Monte Carlo algorithm B' solving $\mathcal{C}$-SAT in time $2^{(p-\epsilon')n}$.*

Here, $\epsilon' > 0$ if $\epsilon > 0$.

▶ **Theorem 12** (Hertli [2])**.** *There exists a Monte-Carlo algorithm solving Unique-3-SAT in time $O\left(2^{(s_3-\epsilon)n}\right)$ for some $\epsilon > 0$.*

Together with Theorem 11 we immediately obtain improvement for general 3-SAT and achieve the currently best running time.

▶ **Theorem 13.** *There is a Monte-Carlo algorithm solving 3-SAT in time $O\left(2^{(s_3-\epsilon')n}\right)$ for some $\epsilon' > 0$.*

2 Proof of Theorem 10

In addition to $Q(\pi,\alpha) = \frac{1}{n!}\cdot 2^{-I(\pi,\alpha)}$, we consider another distribution R on $\mathrm{Sym}(V)\times\mathrm{sat}(F)$. We estimate the success probability of RandomDecode:

$$\begin{aligned}
\Pr[\text{success}] &= \sum_{\alpha\in\mathrm{sat}(F)} \mathop{\mathbb{E}}_{\pi}\left[2^{-C(\pi,\alpha)}\right] = \sum_{\alpha\in\mathrm{sat}(F)} \frac{1}{n!}\sum_{\pi} 2^{-I(\pi,\alpha)-J(\pi,\alpha)} \\
&= \sum_{\alpha\in\mathrm{sat}(F)}\sum_{\pi} R(\pi,\alpha)\frac{2^{-I(\pi,\alpha)-J(\pi,\alpha)}}{n!R(\pi,\alpha)} \\
&= \mathop{\mathbb{E}}_{(\pi,\alpha)\sim R}\left[\frac{Q(\pi,\alpha)}{R(\pi,\alpha)}\cdot 2^{-J(\pi,\alpha)}\right] \\
&\geq 2^{\mathbb{E}_R\left[-\log_2\left(\frac{R(\pi,\alpha)}{Q(\pi,\alpha)}\right)-J(\pi,\alpha)\right]} && (\text{ by Jensen's inequality}) \\
&= 2^{-D(R||Q)-\mathbb{E}_R[J(\pi,\alpha)]}\ ,
\end{aligned}$$

where $D(R||Q)$ is called the *Kullback-Leibler divergence from Q to R*. We can now plug in any distribution R and aim to minimize the expression

$$D(R||Q) + \mathop{\mathbb{E}}_{(\pi,\alpha)\sim R}[J(\pi,\alpha)]\ . \tag{1}$$

Here we face a tradeoff. If we choose R to be uniform over $\mathrm{Sym}(V)\times\mathrm{sat}(F)$, we get $\mathbb{E}_R[J(\pi,\alpha)] = \mathbb{E}_\alpha\left[\sum_x \mathbb{E}_\pi[J_x(\pi,\alpha)]\right] \leq pn$, since P has error at most p; however, $D(R||Q)$ might be too large. Choosing $R = Q$ makes $D(R||Q) = 0$, but the second term can become larger than pn. Informally speaking, the problem is that for certain F, P, and α, if we sample π from the conditional distribution $Q|\alpha$, frozen variables x tend to come earlier (compared to a uniformly sampled π). Thus, when we call $P(F|_\beta, x)$, we have less information (β tends to be a shorter partial assignment), and J_x is more likely to be 1. In Section B we provide examples where these phenomena actually happen.

The process Sample-R below defines a distribution R on pairs (π,α) that resembles Q (and thus keeps the divergence $D(R||Q)$ small) while showing a moderate preference for moving frozen variables to the back of π (keeping $\mathbb{E}_R[J(\pi,\alpha)]$ small). Note that unlike under Q, the marginal distribution on permutations induced by R is not necessarily uniform. Indeed, if we call Sample-R(F,V) for $F = x$ and $V = \{x,y\}$ then π is (y,x) with probability $2/3$ and (x,y) with probability $1/3$. On the other hand, R and Q induce the same marginal distribution on satisfying assignments. The reader is encouraged to verify this, but this property is not required for the proof. We call the resulting distribution R_F to highlight its dependency on F. If F is understood from the context, we simply write R.

▶ **Lemma 14.** *$D(R||Q) \leq p^*\,\mathbb{E}_R[I]$ for every F.*

This is where the mysterious $p^* = \left(\frac{2-\log(e)}{2}\right)$ comes from. The proof of Lemma 14 is a little bit technical but rather straightforward for somebody familiar with information theory, and can be found in the appendix.

▶ **Lemma 15.** *Let $\mathcal{C}$ be a formula class closed under restrictions, P a monotone proof heuristic with error at most p against $\mathcal{C}$. Then for every $F\in\mathcal{C}$ and every frozen variable x of F it holds that $\mathbb{E}_R[J_x] \leq p$.*

This lemma is in some way the heart of our proof. Its proof studies how the conditional distribution $R(\pi|\alpha)$ differs from the uniform distribution over π and applies two careful

Algorithm 4 Sampling from the distribution R

```
1: procedure SAMPLE-R(F, V)
2:     if V = ∅ then
3:         return (∅, ∅)
4:     end if
5:     S(F) := {(x, b) ∈ V × {0, 1} | F|_{x=b} is satisfiable }
6:     (x, b) := a random element from S
7:     (π, α) := SAMPLE-R(F|_{x=b}, V \ {x})
8:     return (xπ, α ∪ [x = b])
9: end procedure
```

coupling arguments. It is also the place where we use that P is monotone. Lemma 15 has the following consequence:

▶ **Lemma 16.** *$\mathbb{E}_R[pI_x + J_x] \le p$ for every $x \in V$, and $\mathbb{E}_R[pI + J] \le pn$.*

Proof. Imagine we run the process SAMPLE-R but pause when (1) x becomes frozen or (2) x, as a non-frozen variable, is chosen in line 6. Everything what happens before the pause is called *the past*. If (2) happens, then $I_x = 1, J_x = 0$ and thus $\mathbb{E}_R[pI_x + J_x|\text{the past}] = \mathbb{E}_R[p \cdot 1 + 0] = p$. Otherwise, if (1) happens, then $I = 0$ since x becomes frozen, and $\mathbb{E}_R[pI_x + J_x|\text{the past}] = \mathbb{E}_R[J_x|\text{the past}]$. After *the past* has happened, the sampling process has arrived at a new formula $F' \in \mathcal{C}$, and x is frozen in F'. Since $\mathcal{C}$ is closed under restrictions, $F' \in \mathcal{C}$, too, and we can apply Lemma 15 to conclude that $\mathbb{E}_{R_F}[J_x|\text{the past}] = \mathbb{E}_{R_{F'}}[J_x] \le p$. Thus, $\mathbb{E}_R[pI_x + J_x] \le p$. ◀

▶ **Lemma 17.** *$\mathbb{E}_R[I] = \mathbb{E}_Q[I]$.*

Let us put everything together. $D(R||Q) + \mathbb{E}_R[J] \le p^* \mathbb{E}_R[I] + \mathbb{E}_R[J] = \mathbb{E}_R[pI + J] - (p - p^*)\mathbb{E}_R[I] \le pn - q\mathbb{E}_Q[I]$. Thus, RANDOMDECODE succeeds with probability at least $2^{-pn+q\,\mathbb{E}_Q[I]}$. This proves Theorem 10.

3 Unique to General

We are now ready to prove Theorem 11, which claims that if you can beat PPSZ for UNIQUE-$\mathcal{C}$-SAT, then you can beat it for $\mathcal{C}$-SAT.

Proof of Theorem 11. Let $\delta > 0$ be a fixed number, to be determined later. If $\mathbb{E}_Q[I] \ge \delta \cdot n$, then

$$\Pr[\text{RANDOMDECODE}(F, P) \text{ successful}] \ge 2^{-pn+\delta cn}\,, \tag{2}$$

which is exponentially larger than 2^{-pn}.

Otherwise, assume that $\mathbb{E}_Q[I] \le \delta n$. In particular, $I(\pi, \alpha) \le \delta n$ for *some* permutation π and assignment α. This means that there is a partial assignment ρ fixing δn variables such that $F|\rho$ has a unique satisfying assignment. We prove Point 1 of the theorem. When running RANDOMDECODE on F, with probability $\binom{n}{\le \delta n}^{-1} \cdot 2^{\delta n}$ the first δn steps produce exactly ρ, and the remaining $(1-\delta)n$ steps are like running RANDOMDECODE$(F|_\rho, P)$. $F|_\rho$, has the unique solution α, and thus RANDOMDECODE$(F|_\rho, P)$ finds α with probability at least $2^{(-p+\epsilon)(n-\delta)}$. Altogether,

$$\Pr[\text{RANDOMDECODE}(F, P) = \alpha] \ge \binom{n}{\delta n}^{-1} \cdot 2^{-\delta n} \cdot 2^{(-p+\epsilon)(n-\delta)}\,. \tag{3}$$

By choosing $\delta > 0$ optimally, we can make sure that both (2) and (3) are at least $2^{(-p+\epsilon')n}$, for some $\epsilon' > 0$. This proves Point 1 of the theorem. The proofs of the other two points are similar. ◀

4 Open Questions

Can we show that formulas with a unique solution are the worst case for RandomDecode under every "reasonable" heuristic P?

Can we show that the success probability of RandomDecode is exponentially larger than 2^{-pn} if F has an exponential number of solutions? Unfortunately, the current "bonus term" $\mathbb{E}_Q[I]$" can be *constant* for some formulas with a large number of solutions, for example for $F = (x_1 \wedge \cdots \wedge x_{n/2}) \vee (|\mathbf{x}| \leq 100)$ (note that $\mathbb{E}_Q[I]$ only depends on the underlying boolean function, not on its representation as a CNF formula).

Acknowledgements. Dominik Scheder want to thank Navid Talebanfard for inspiring discussions.

References

1. Timon Hertli. 3-SAT faster and simpler – unique-SAT bounds for PPSZ hold in general. In *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science – FOCS 2011*, pages 277–284. IEEE Computer Soc., Los Alamitos, CA, 2011. doi:10.1109/FOCS.2011.22.
2. Timon Hertli. Breaking the PPSZ Barrier for Unique 3-SAT. In Javier Esparza, Pierre Fraigniaud, Thore Husfeldt, and Elias Koutsoupias, editors, *Automata, Languages, and Programming – 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part I*, volume 8572 of *Lecture Notes in Computer Science*, pages 600–611. Springer, 2014. doi:10.1007/978-3-662-43948-7_50.
3. Timon Hertli, Robin A. Moser, and Dominik Scheder. Improving PPSZ for 3-SAT using critical variables. In *Proceedings of STACS*, pages 237–248, 2011. URL: http://arxiv.org/abs/1009.4830.
4. 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.
5. Kazuo Iwama, Kazuhisa Seto, Tadashi Takai, and Suguru Tamaki. Improved randomized algorithms for 3-SAT. In *Algorithms and Computation*, volume 6506 of *Lecture Notes in Comput. Sci.*, pages 73–84. Springer Berlin / Heidelberg, 2010.
6. Kazuo Iwama and Suguru Tamaki. Improved upper bounds for 3-SAT. In *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 328–329 (electronic), New York, 2004. ACM.
7. Ramamohan Paturi, Pavel Pudlák, Michael E. Saks, and Francis Zane. An improved exponential-time algorithm for k-SAT. *J. ACM*, 52(3):337–364 (electronic), 2005. doi:10.1145/1066100.1066101.
8. Ramamohan Paturi, Pavel Pudlák, and Francis Zane. Satisfiability coding lemma. *Chicago J. Theoret. Comput. Sci.*, pages Article 11, 19 pp. (electronic), 1999.
9. Daniel Rolf. Improved Bound for the PPSZ/Schöning-Algorithm for 3-SAT. *Journal on Satisfiability, Boolean Modeling and Computation*, 1:111–122, 2006.
10. Dominik Scheder, Bangsheng Tang, Shiteng Chen, and Navid Talebanfard. Exponential lower bounds for the PPSZ k-sat algorithm. In Sanjeev Khanna, editor, *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013,*

New Orleans, Louisiana, USA, January 6-8, 2013, pages 1253–1263. SIAM, 2013. doi:10.1137/1.9781611973105.91.

11 Uwe Schöning. A probabilistic algorithm for k-SAT and constraint satisfaction problems. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science*, pages 410–414. IEEE Computer Society, Los Alamitos, CA, 1999. doi:10.1109/SFFCS.1999.814612.

A Proof of the lemmas

For a formula F over variable set V, recall that $S(F,V)$ is the set of all pairs $(x,b) \in V \times \{0,1\}$ for which $F|_{x=b}$ is satisfiable. Note that if F is satisfiable then $|S(F,V)|$ is n plus the number of liquid variables.

▶ **Lemma 14** (restated). *$D(R||Q) \leq \left(\frac{2-\log(e)}{2}\right) \mathbb{E}_R[I]$, for every formula F.*

Proof. Let us spell out a pair (π,α) as $(x_1 \dots x_n, b_1 \dots b_n)$, where x_i is the i^{th} variable under π and $b_i = \alpha(x_i)$. Let $\tau_i := (x_1 \dots x_i, b_1 \dots b_i)$ be a "prefix" of (π,α). Define R_{τ_i} be the distribution of (b_{i+1}, x_{i+1}) under R conditioned on τ_i. Similarly define Q_{τ_i}. By the chain rule for the divergence we get

$$D(R||Q) = \sum_{i=0}^{n-1} \mathop{\mathbb{E}}_{\tau_i \sim R}[D(R_{\tau_i}||Q_{\tau_i}) \ .$$

So let us fix a "past" τ_i and bound $D(R_{\tau_i}||Q_{\tau_i})$. Let $F_i := F|_{x_1 \mapsto b_1 \dots x_i \mapsto b_i}$ and $V_i := \{x_{i+1}, \dots, x_n\}$. So F_i is a CNF formula over V_i, and it is exactly the formula for which Sample-R is called in its i^{th} call. Let $n_i = |V_i|$, $s_i := |S(F_i,V_i)|$, f_i the number of frozen variables in V_i and l_i the number of liquid variables. Thus $f_i + l_i = n_i$ and $f_i + 2l_i = s_i$. Note that R_i is uniform over $S(F_i,V_i)$. Q_{τ_i} picks x_{i+1} uniformly at random from V_i and assigns it a random value from the (one or two) allowed values. Thus, $Q_{\tau_i}(x,b)$ is 0 if $(x,b) \notin S(F_i,V_i)$; otherwise, it is $1/n_i$ if x is frozen and $1/2n_i$ if x is liquid.

$$\begin{aligned}
D(R_{\tau_i}||Q_{\tau_i}) &= \sum_{(x,b)\in S(F_i,V_i)} R_{\tau_i}(x,b) \log\left(\frac{R_{\tau_i}(x,b)}{Q_{\tau_i}(x,b}\right) \\
&= \sum_{(x,b)\in S(F_i,V_i)} \frac{1}{s_i} \log\left(\frac{1/s_i}{[1/n_i \text{ if } x \text{ frozen, } 1/2n_i \text{ if } x \text{ liquid }]}\right) \\
&= \frac{2l_i}{s_i}\log\left(\frac{1/s_i}{1/2n_i}\right) + \frac{f_i}{s_i}\log\left(\frac{1/s_i}{1/n_i}\right) = \frac{2l_i}{s}\log\left(\frac{2n_i}{s_i}\right) + \frac{f_i}{s_i}\log\left(\frac{n_i}{s_i}\right) \\
&= \frac{2l_i}{s} + \log\left(\frac{n_i}{s_i}\right) = \frac{2l_i}{s} + \log\left(1 - \frac{l_i}{s_i}\right) \\
&\leq \frac{2l_i}{s} - \log(e)\frac{l_i}{s_i} = \frac{l_i}{s_i}(2 - \log(e)) \ .
\end{aligned}$$

Let $\tilde{I}_i(\pi,\alpha) := I_{x_i}(\pi,\alpha)$, i.e., an indicator variable which is 1 if the i^{th} variable under π is liquid in F_{i-1}. We observe that $\mathbb{E}_{R_{\tau_i}}[\tilde{I}_{i+1}] = \frac{2l_i}{s_i}$, since there are exactly $2l_i$ pairs $(x,b) \in S(F_i,V_i)$ for which the variable x is liquid. Putting everything together, we get

$$D(R||Q) \leq \sum_{i=0}^{n-1} \mathop{\mathbb{E}}_{\tau_i \sim R}\left[\frac{l_i}{s_i}(2 - \log(e))\right] = \frac{2 - \log(e)}{2} \sum_{i=0}^{n-1} \mathop{\mathbb{E}}_{\tau_i \sim R}\left[\frac{2l_i}{s_i}\right] \ .$$

As we have just seen, the latter sum equals $\mathbb{E}_R\left[\sum_{i=1}^{n} \tilde{I}_i\right]$, which again equals $\mathbb{E}_R[I]$, since $\tilde{I}_i$, $i = 1, \dots, n$ simply counts I_x, $x \in V$ in a different order. ◀

A.1 Permutations that delay x – Proof of Lemma 15

Before we prove Lemma 15, we have to introduce some notation. We call a function $g : 2^V \to \mathbf{R}$ *monotone* if $g(A) \leq g(B)$ for any $A \subseteq B \subseteq V$. Let $x \in V$ be a fixed variable, $\pi \in \mathrm{Sym}(V)$ a permutation. We denote by $W(\pi)$ the set of variables appearing after x in π. Observe that $J_x(\pi, \alpha)$ only depends on $W(\pi)$, not on the particular order of the variables coming before x and of those coming after x.

▶ **Observation 18.** *J_x is a monotone function in W, since P is a monotone heuristic.*

For two strings σ, π, we write $\sigma \preceq \pi$ if σ is a prefix of π. A permutation π on set V of size n can be viewed as a string in V^n without repeated letters. A string $\sigma \in V^*$ without repeated letters is called a *partial permutation*. If D is a distribution over permutations on V and σ is a partial permutation, we write $D(\sigma) := \Pr_{\pi \sim D}(\sigma \preceq \pi)$.

▶ **Definition 19.** Let D be a distribution over permutations on V, and let $x \in V$. We say D *delays* x if for all $y \in V$ and all partial permutations σ not containing x or y, it holds that $D(\sigma x) \leq D(\sigma y)$.

Informally, at every stage, x is among the least likely elements to come next. For example, the uniform distribution delays x; so does the distribution that samples a permutation of $V \setminus \{x\}$ and places x at the end. Lemma 15 will follow from the next two lemmas:

▶ **Lemma 20.** *The distribution $(R|\alpha)$ delays x, for every frozen variable x.*

Here, $(R|\alpha)$ is the distribution on permutations conditioned on this fixed satisfying assignment α, i.e., $(R|\alpha)(\pi) = R(\pi, \alpha|\alpha)$.

▶ **Lemma 21.** *Let V be a finite set, $x \in V$, D a distribution over permutations of V that delays x, and $f : V \to \mathbb{R}$ a monotone function. Denote by $W = W(\pi)$ the set of elements coming after x in π. Then*

$$\mathop{\mathbb{E}}_{\pi \sim D}[f(W)] \leq \mathop{\mathbb{E}}_{\pi \sim \mathcal{U}}[f(W)] \ ,$$

where $\mathcal{U}$ is the uniform distribution over permutations.

Proof Idea. Since D delays x, the set W tends to be smaller under D than under $\mathcal{U}$. Since f is monotone this means the expectation $f(W)$ is smaller, too. This is the intuition. The formal proof uses a coupling argument. ◀

▶ **Lemma 15** (restated). *Let $\mathcal{C}$ be a formula class closed under restrictions, P a monotone proof heuristic with error at most p against $\mathcal{C}$. Then for every $F \in \mathcal{C}$ and every frozen variable x of F it holds that $\mathbb{E}_R[J_x] \leq p$.*

Proof. By assumption on P we have $\mathbb{E}_\pi[J_x(\pi, \alpha)] \leq p$ when π is uniform. Thus, we have to compare how the uniform distribution and $(R|\alpha)$ differ in their treatment of x, and how $J_x(\pi, \alpha)$ reacts to these differences. By Lemma 20, $(R|\alpha)$ delays x. By Observation 18, J is a monotone function in W, where $W = W(\pi)$ is the set of elements coming after x in π. Thus, by Lemma 21 we obtain that $\mathbb{E}_{\pi \sim R}[J_x(\pi, \alpha)] \leq \mathbb{E}_{\pi \sim \mathcal{U}}[J_x(\pi, \alpha)] \leq p$. ◀

A.2 Remaining proofs – Lemma 20 and Lemma 21

Proof of Lemma 20. By assumption, x is frozen and σ is a partial permutation not containing x nor y. Assume first that σ is empty. We have to show that $R(x|\alpha) \leq R(y|\alpha)$ or, equivalently, $R(x,\alpha) \leq R(y,\alpha)$.[2]

Consider the following alternative but equivalent way to sample R: order the s elements of $S(F,V)$ randomly into a sequence $\tau = (x_1,b_1),\ldots,(x_s,b_s)$ and then add the unit clauses $(x_i = b_i)$ to F, in this order, skipping a unit clause if adding it would make F unsatisfiable. This adds n unit clauses in some order $(x_{i_1} = b_{i_1}),\ldots,(x_{i_n} = b_{i_n})$ and thus defines a permutation π of V and an assignment α. The pair (π,α) has distribution R.

Let $T_{z,\alpha}$ denote the set of all such sequences τ that (1) result in α and (2) place z at the beginning of π. So $R(z,\alpha) = \frac{|T_{z,\alpha}|}{|S(F,V)|!}$. Since the first unit clause $(x_1 = b_1)$ in a sequence is always consistent with F, every sequence in $T_{z,\alpha}$ starts with $(z = \alpha(z))$. For a sequence $\tau \in T_{x,\alpha}$ define $f(\tau)$ to be the sequence τ' where we switch the positions of $(x = \alpha(x))$ and $(y = \alpha(y))$ (note that both must appear in τ, and $(x = \alpha(x))$ appears at the beginning). A minute of thought shows that the sequence $f(\tau)$ leads to α as well (the key observation is that x is frozen, so logically $(x = \alpha(x))$ is already present in F, whether it occurs at the beginning of τ or not). Thus $f(\tau) \in T_{y,\alpha}$ and we have just defined an injection from $T_{x,\alpha}$ into $T_{y,\alpha}$. This shows that $|T_{x,\alpha}| \leq |T_{y,\alpha}|$ and thus $R(x,\alpha) \leq R(y,\alpha)$.

If σ is not empty we write $\alpha = \alpha_\sigma\alpha_{\bar{\sigma}}$, where α_σ is the α restricted to the variables appearing in σ, and $\alpha_{\bar{\sigma}}$ is the rest. Write $F' := F|_{\alpha_\sigma}$ Now $R(\sigma z,\alpha)$ is the probability that Sample-R follows σ and α in its first $|\sigma|$ steps, times $R_{F'}(z,\alpha_{\bar{\sigma}})$. Thus, we have reduced non-empty σ case to the empty σ case. ◀

▶ **Lemma 21** (restated). *Let V be a finite set, $x \in V$, D be a distribution over permutations of V that delays x, and $f : V \to \mathbb{R}$ be a monotone function. Denote by $W = W(\pi)$ the set of elements coming after x in π. Then*

$$\mathop{\mathbb{E}}_{\pi\sim D}[f(W)] \leq \mathop{\mathbb{E}}_{\pi\sim\mathcal{U}}[f(W)] \ ,$$

where $\mathcal{U}$ is the uniform distribution over permutations.

Proof. Let W_D denote a random variable distributed like $W(\pi)$ with $\pi \sim D$, and similarly $W_\mathcal{U} = W(\pi)$ where π is uniform. Below, we define a process Sample-W which simultaneously samples W_D and $W_\mathcal{U}$ and guarantees $W_D \subseteq W_\mathcal{U}$. In other words, Sample-W defines a coupling under which $W_D \subseteq W_\mathcal{U}$ We write $D(z|\sigma) := D(\sigma z|\sigma) = \frac{D(\sigma z)}{D(\sigma)}$. This is the probability that z is chosen next, conditioned on σ having been sampled so far.

The process Sample-W clearly samples W_D from the correct distribution. Note that an element z gets removed from $W_\mathcal{U}$ whenever $t < D(x|\sigma)$, and then a uniformly random element is removed. Also, the process terminates when x has been removed from W_D. Obviously, it will be removed from $W_\mathcal{U}$ in the same iteration. So W_D and $W_\mathcal{U}$ have the correct distribution. Lastly, since $D(x|\sigma) \leq D(z|\sigma)$, when the element z is removed from $W_\mathcal{U}$, it has already been removed from W_D. Thus, $W_D \subseteq W_\mathcal{U}$ holds in every step. Thus, $f(W_D) \leq f(W_\mathcal{U})$ with probability 1 and therefore $\mathbb{E}_{\pi\sim D}[f(W)] \leq \mathbb{E}_{\pi\sim\mathcal{U}}[f(W)]$. ◀

[2] We have not formally introduced this notation. It is the probability that Sample-R outputs α and a permutation π starting with x (respectively, y)

Algorithm 5 Sampling W_D and $W_{\mathcal{U}}$

```
1: procedure SAMPLE-W(V, x)
2:     σ := the empty string
3:     W_D = W_U = V
4:     while x ∈ W_D do
5:         (z, t) ∈ V × [0, 1], uniformly at random
6:         if t < D(z|σ) and z ∈ W_D then
7:             remove z from W_D
8:             append z to σ
9:         end if
10:        if t < D(x|σ) then
11:            remove z from W_U
12:        end if
13:    end while
14:    return W_D, W_U
15: end procedure
```

B Bad Examples

B.1 Why s Direct Application of Jensen's Does Not Work

We will demonstrate why proving Theorem 6 requires nontrivial effort. Let us proceed as in the proof of Observation 5. Let $\mathrm{sat}(F)$ be the set of all satisfying assignments of F. The success probability of Decode is

$$\begin{aligned}\Pr_{c,\pi}[\text{success}] &= \sum_{\alpha\in\mathrm{sat}(F)} \Pr_{c,\pi}[\textsc{Decode}(c,\pi,F,P)=\alpha] \\ &= \sum_{\alpha\in\mathrm{sat}(F)} \mathop{\mathbb{E}}_{\pi}\left[2^{-C(\pi,\alpha)}\right] \qquad (4)\\ &\geq \sum_{\alpha\in\mathrm{sat}(F)} 2^{-\mathbb{E}_\pi[C(\pi,\alpha)]}\,, \qquad (5)\end{aligned}$$

where last line follows from Jensen's inequality.

We will construct an example in which $\Pr[\text{success}] = 1$ but (5) is exponentially small. Consider $P = P_\infty$, the complete proof heuristic, which has error 0 against, well, every circuit class. Also note that (4) is 1, as Decode always returns a satisfying assignment if given access to P_∞. Let F be the Boolean function defined by $F(x) = 1$ if $|x| = 1$, i.e., exactly one of the n positions of x is 1. So $\mathrm{sat}(F) = \{\mathbf{e}_1, \ldots, \mathbf{e}_n\}$. Note that since P_∞ is the complete prover, it does not really matter in which way we represent F.

By symmetry, $\Pr[\textsc{Decode}(c,\pi,F) = \mathbf{e}_i] = 1/n$ for every i. What is $C(\mathbf{e}_i,\pi)$? Let j be the position of x_i in π. A minute of thought shows that $C(\mathbf{e}_i,\pi) = \min(j, n-1)$. Therefore

$$\mathop{\mathbb{E}}_{\pi}[C(\mathbf{e}_i,\pi)] = \frac{1}{n}\cdot\sum_{j=1}^{n-1} j + \frac{1}{n}(n-1) \geq \frac{\binom{n}{2}}{n} = \frac{n-1}{2}\,.$$

Summing up over all $\mathrm{sat}(F)$ we see that

$$(5) = \sum_{\alpha\in\mathrm{sat}(F)} 2^{-\mathbb{E}_\pi[C(\pi,\alpha)]} \leq n\cdot 2^{-\frac{n-1}{2}}\,.$$

Thus, there is an exponential gap between (5) and $2^{-pn} = 2^{-0 \cdot n} = 1$, the bound in the conjecture. We conclude that this "naive" application of Jensen's inequality will not work.

B.2 A Smarter Application of Jensen's Inequality

Suppose we run $\textsc{Decode}(c, \pi, F)$ with random c and π and the complete prover P_∞. It will always return a satisfying assignment, and thus defines a probability distribution Q over $\mathrm{Sym}(V) \times \mathrm{sat}(F)$. It is easy to see that

$$Q(\pi, \alpha) = Q(\pi) \cdot Q(\alpha|\pi) = \frac{1}{n!} \cdot 2^{-I(\pi,\alpha)} \ .$$

We can now rewrite the success probability of Decode (using some incomplete proof heuristic P) as

$$\begin{aligned} \Pr[\text{success}] &= \sum_{\alpha \in \mathrm{sat}(F)} \mathop{\mathbb{E}}_{\pi} \left[2^{-C(\pi,\alpha)}\right] = \sum_{\pi,\alpha} \frac{1}{n!} 2^{-I(\pi,\alpha)-J(\pi,\alpha)} \\ &= \mathop{\mathbb{E}}_{(\pi,\alpha)\sim Q} \left[2^{-J}\right] \qquad (6) \\ &\geq 2^{-\mathbb{E}_Q[J]} \ . \qquad (7) \end{aligned}$$

Sadly, (7) can be exponentially smaller than 2^{-pn}, as we will show now.

B.3 Another Bad Example

Consider the following function:

$$\textsc{Exactly-Two}(x, y, z) \wedge \bigwedge_{i=1}^{n} (\textsc{At-Least-Two}(x, y, z) \to a_i) \, .$$

We can express this as a 3-CNF formula:

$$\begin{aligned} &(x \vee y) \wedge (x \vee z) \wedge (y \vee z) \wedge (\bar{x} \vee \bar{y} \vee \bar{z}) \wedge \\ &\bigwedge_{i=1}^{n} ((\bar{x} \vee \bar{y} \vee a_i) \wedge (\bar{x} \vee \bar{z} \vee a_i) \wedge (\bar{y} \vee \bar{z} \vee a_i)) \ . \end{aligned}$$

Enumerating our variables as $x, y, z, a_1, \ldots, a_n$, the satisfying assignments are $\alpha_1 = (0111^n)$, $\alpha_2 = (1011^n)$, and $\alpha_3 = (1101^n)$. Consider the prover $P = P_1$, i.e., it checks whether the variable in question is contained in a unit clause. Since this is a 3-CNF, the error probability of P is at most $2/3$. What is $\mathbb{E}_Q[J]$?

$$\begin{aligned} \mathop{\mathbb{E}}_{Q}[J] &= \mathop{\mathbb{E}}_{\alpha\sim Q}[\mathop{\mathbb{E}}_{\pi\sim Q|\alpha}[J]] = \mathop{\mathbb{E}}_{\pi\sim Q|\alpha_1}[J(\alpha_1, \pi)] && \text{(by symmetry between the } \alpha_i) \\ &\geq n \mathop{\mathbb{E}}_{\pi\sim Q|\alpha_1}[J_{a_1}(\alpha_1, \pi)] \ . && \text{(by symmetry between the } a_i) \end{aligned}$$

One can now show by a straightforward calculation that $\mathbb{E}_{\pi\sim Q|\alpha_1}[J_{a_1}(\alpha_1, \pi)] = \frac{11}{16} > 2/3$. Thus, the expression $2^{-E_Q[J]}$ can be exponentially smaller than $2^{-\frac{2}{3}\cdot n}$, which is the true worst-case success probability of PPZ (i.e., PPSZ with proof heuristic P_1) on 3-CNF formulas. We strongly encourage the reader to compute $\mathbb{E}_{\pi\sim Q|\alpha_1}[J_{a_1}(\alpha_1, \pi)]$ for the above example.

Problem Assessment

Since π is uniform under Q, it holds that $Q(\pi|\alpha)$ is proportional to $Q(\alpha|\pi) = 2^{-I(\pi,\alpha)}$. For $\alpha_1 = (0111^n)$, the latter term is largest when x comes first (as setting x to 0 implies the values of both y and z). Informally speaking, y and z tend to come later among x, y, z. When can P_1 tell the value of a_1? The clause $(\bar{y} \vee \bar{z} \vee a_1)$ reduces to the unit clause (a_1) if y, z come before a_1. Normally, this happens with probability 1/3. Under $Q|\alpha_1$, however, y and z tend to come later, and the probability decreases to 5/16, and thus $\mathbb{E}_{\pi\sim Q|\alpha_1}[J_{a_1}(\alpha_1, \pi)] = 11/16$.

Noise Stability Is Computable and Approximately Low-Dimensional*†

Anindya De[1], Elchanan Mossel[2], and Joe Neeman[3]

1 **Northwestern University, Evanston, IL, USA**
de.anindya@gmail.com
2 **Massachusetts Institute of Technology, Cambridge, MA, USA**
elmos@mit.edu
3 **University of Bonn, Bonn, Germany; and**
University of Texas at Austin, Austin, TX, USA
joeneeman@gmail.com

Abstract

Questions of noise stability play an important role in hardness of approximation in computer science as well as in the theory of voting. In many applications, the goal is to find an optimizer of noise stability among all possible partitions of $\mathbb{R}^n$ for $n \geq 1$ to k parts with given Gaussian measures $\mu_1, \ldots, \mu_k$. We call a partition ϵ-optimal, if its noise stability is optimal up to an additive ϵ. In this paper, we give an explicit, computable function $n(\epsilon)$ such that an ϵ-optimal partition exists in $\mathbb{R}^{n(\epsilon)}$. This result has implications for the computability of certain problems in non-interactive simulation, which are addressed in a subsequent work.

1998 ACM Subject Classification F.2 Analysis of Algorithms and Problem Complexity

Keywords and phrases Gaussian noise stability, Plurality is stablest, Ornstein Uhlenbeck operator

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.10

1 Introduction

1.1 Isoperimetric Theory and Noise Stability

Isoperimetric problems have been studied in mathematics since antiquity. The solution to the isoperimetric problem in the two dimensional Euclidean plane was known in ancient Greece. The study of isoperimetric problems is central in modern mathematics and theoretical computer science. Some central examples include the study of expanders and mixing of Markov chains.

Our interest in this work is in a central modern isoperimetric problem, i.e, the problem of noise stability. This problem, originally studied by Borell in relation to the study of the heat equation in mathematical physics [3], has emerged as a central problem in theoretical computer science [13], as well as in combinatorics and in voting theory [11] as we elaborate below.

The fundamental question in this area is to find a partition of Gaussian space (with prescribed measures) which maximizes the noise stability of the partition. We equip $\mathbb{R}^n$

* Full version of this paper is available at https://arxiv.org/abs/1701.01483.
† E. M. acknowledges the support of grant N00014-16-1-2227 from Office of Naval Research and of NSF award CCF 1320105. A. D. acknowledges the support of a start-up grant from Northwestern University.

with the standard Gaussian measure γ_n, i.e. the measure with density $(2\pi)^{-n/2}\exp(-|x|^2/2)$, where $|x|$ denotes the Euclidean norm. The Ornstein-Uhlenbeck operator P_t is defined for $t \in [0,\infty)$ and (sufficiently integrable) $f : \mathbb{R}^n \to \mathbb{R}$ by

$$(P_t f)(x) = \int_{y\in\mathbb{R}^n} f(e^{-t}\cdot x + \sqrt{1-e^{-2t}}\cdot y)d\gamma_n(y).$$

The resulting notion of noise stability is defined by $\mathsf{Stab}_t(f) := \mathbf{E}[fP_tf]$, where the expectation is with respect to γ_n. If f denotes the indicator function of a set (call it $\mathcal{A}_f$), then $\mathsf{Stab}_t(f)$ is the probability that two e^{-t}-correlated Gaussian random variables x, y both fall in $\mathcal{A}_f$. In the limit $t \to 0$, noise stability captures the Gaussian surface area of the set $\mathcal{A}_f$ [15]. In particular, we have the following relation for any set:

$$\mathsf{GAS}(\mathcal{A}_f) = \lim_{t\downarrow 0} \frac{\sqrt{2\pi}}{\arccos(e^{-t})}\cdot \mathbf{E}[f\cdot(f-P_tf)],$$

where $\mathsf{GAS}(\mathcal{A}_f)$ denotes the Gaussian surface area of the set $\mathcal{A}_f$.

1.2 Halfspaces are most stable sets

For both noise stability and surface area, halfspaces are known to be the optimal sets [2, 20, 3, 1]. We will state this fact in a slightly convoluted way, in order to more easily generalize it. Let Δ_k denote the probability simplex in $\mathbb{R}^k$ (i.e. the convex hull formed by the standard unit vector $\{\mathbf{e}_1,\ldots,\mathbf{e}_k\}$). Any function with range $[k] := \{1,\ldots,k\}$ naturally embeds in Δ_k by identifying $i \in [k]$ with $\mathbf{e}_i$. Moreover, we extend the Ornstein-Uhlenbeck operator to act on vector valued functions in the obvious way: if $f = (f_1,\ldots,f_k) : \mathbb{R}^n \to \mathbb{R}^k$ then $P_tf = (P_tf_1,\ldots,P_tf_k)$. Finally, say that $f = (f_1, f_2) : \mathbb{R}^n \to \Delta_2$ is a *halfspace* if there exist $a, b \in \mathbb{R}^n$ such that $f_1(x) = 1_{\{\langle x-a,b\rangle\le 0\}}$, where $\langle\cdot,\cdot\rangle$ denotes the Euclidean inner product.

▶ **Theorem 1** (Borell). *For any $g : \mathbb{R}^n \to \Delta_2$, a halfspace $f : \mathbb{R}^n \to \Delta_2$ with $\mathbf{E}[f] = \mathbf{E}[g]$ satisfies*

$$\mathbf{E}[\langle f, P_tf\rangle] \ge \mathbf{E}[\langle g, P_tg\rangle].$$

Theorem 1 has many applications in computational complexity. Most famously, it can be combined with an *invariance principle* [16] in order to prove a number of tight hardness-of-approximation results under the unique games conjecture [12].

1.3 More parts?

It is straightforward to extend the notion of noise stability to partitions with many parts. Namely, a partition of $\mathbb{R}^n$ can be described by $f : \mathbb{R}^n \to [k]$. Identifying $[k]$ with $\{\mathbf{e}_1,\ldots,\mathbf{e}_k\}$ as above, we define the noise stability of such a partition by $\mathbf{E}[\langle f, P_tf\rangle]$. One may then ask for an analogue of Theorem 1, but where Δ_2 is replaced by Δ_k for some $k \ge 3$.

Even the three-part version of Theorem 1 turns out to be rather harder than the two-part one. We will try to explain why, by analogy with isoperimetric problems. First of all, in the limit as $t \downarrow 0$, the Euclidean analogue of Theorem 1 for $k = 3$ is known as the "double bubble" problem. The well-known "double bubble conjecture" states that the minimum total surface area of two bodies separating and enclosing two given (Lebesgue) volumes is achieved by two spheres meeting at 120°. After being open for more than a century, this problem was settled rather recently in $\mathbb{R}^2$, $\mathbb{R}^3$, and $\mathbb{R}^4$ [9, 10, 19]. For the Gaussian space, [4] showed that for some small but positive constant $c > 0$, the Gaussian surface area of three partitions is

minimized by the "standard simplex partition" as long as the measures of all the three parts is within $1/3 \pm c$; a standard simplex partition is one with three flat boundaries which meet at a single point at 120° angles.

Since halfspaces both maximize noise stability and minimize Gaussian surface area, and since the standard simplex partition is known to minimize multi-part Gaussian surface area in certain cases, it seems natural to guess that the standard simplex partition also maximizes multi-part noise stability. This was explicitly conjectured in [12], in the special case that all of the parts in the partition have equal measure. However, a somewhat surprising (at least to the authors) recent work [7] showed that the standard simplex partition fails to maximize multi-part noise stability *unless* all of the parts have equal measure. On the other hand, there is also some support for the conjecture in the equal-measure case: Heilman [8] showed that the conjecture is true in $\mathbb{R}^n$ for $n \leq n_0(t)$.

1.4 Approximate noise stability of multipartitions?

In light of the uncertainty about optimal partitioning for $k \geq 3$, one can ask a more modest question. Given $k \geq 3$, $t > 0$, and prescribed measures for the k parts, let α_n be the optimal noise stability that can be obtained in $\mathbb{R}^n$ under these constraints. Since the Gaussian measure is a product measure, α_n is clearly non-increasing in n. Since it is bounded below by zero, it has a limit as $n \to \infty$.

Our main result is that there is an explicitly computable $n_0 = n_0(k, t, \epsilon)$ such that $\alpha_{n_0} \geq \alpha_m - \epsilon$ for all $m \in \mathbb{N}$. Although the bound on n_0 that we give is not particularly good, the key point is that it is explicitly computable. As a consequence, up to error ϵ, the noise stable partition is also explicitly computable. We conclude the introduction by an open question:

▶ **Question.** *Does there exists n_0 such that $\alpha_{n_0} = \alpha_n$ for $n > n_0$?*

Our current techniques are not suitable for addressing the question above.

2 Main theorem and overview of proof technique

In order to state the main theorem, we first need to recall the notion of a polynomial threshold function. A function $f : \mathbb{R}^n \to \{0, 1\}$ is said to be a degree-d PTF if there exists a polynomial $p : \mathbb{R}^n \to \mathbb{R}$ of degree d such that $f(x) = 1$ if and only if $p(x) > 0$. We will need a k-ary generalization of this definition. We note that there are several possible ways to generalize the notion of PTFs to k-ary PTFs and our particular choice is dictated by the convenience of using the relevant results from [5].

▶ **Definition 2.** A function $f : \mathbb{R}^n \to [k]$ is said to be a multivariate PTF if there exist polynomials $p_1, \ldots, p_k : \mathbb{R}^n \to \mathbb{R}$ such that

$$f(x) = \begin{cases} j & \text{if } p_j(x) > 0 \text{ and for } i \neq j,\ p_i(x) \leq 0\,, \\ 1 & \text{otherwise}\,. \end{cases}$$

In this case, we denote $f = \mathsf{PTF}(p_1, \ldots, p_k)$. Further, f is said to be a degree-d multivariate PTF if $p_1, \ldots, p_k$ are degree d polynomials.

We now state the main theorem of this paper. We set the convention, that unless explicitly mentioned otherwise, the underlying distribution is γ_n, the standard n-dimensional Gaussian measure. Likewise, given any random variable X over $\mathbb{R}^k$, $\mathbf{E}[X]$ denotes its (vector-valued) expectation and $\mathsf{Var}(X)$ denotes its covariance matrix.

▶ **Theorem 3.** *Let $f : \mathbb{R}^n \to [k]$ such that $\mathbf{E}[f] = \mu \in \mathbb{R}^k$. Then, given any $t > 0, \epsilon > 0$, there exists an explicitly computable $n_0 = n_0(t, k, \epsilon)$ and $d = d(t, k, \epsilon)$ such that there is a degree-d PTF $g : \mathbb{R}^{n_0} \to [k]$ such that:*

1. $\|\mathbf{E}[f] - \mathbf{E}[g]\|_1 \leq \epsilon$,
2. $\mathbf{E}[\langle g, P_t g\rangle] \geq \mathbf{E}[\langle f, P_t f\rangle] - \epsilon$.

Note that the above theorem automatically implies that a function g satisfying the above properties can be explicitly computed (up to some additional error ϵ). This is because the set of degree-d PTFs on $\mathbb{R}^{n_0}$ admits a finite sized explicitly enumerable ϵ-cover.

2.1 From general partitions to PTF

The first step in the proof of Theorem 3 is to show that given any $f : \mathbb{R}^n \to [k]$, there is a multivariate PTF $g' : \mathbb{R}^n \to [k]$ which meets the two criteria in Theorem 3 and has degree $d = d(t, k, \epsilon)$, for some explicit function $d(t, k, \epsilon)$. In other words, g' satisfies $\|\mathbf{E}[f] - \mathbf{E}[g']\|_1 \leq \epsilon$ and $\mathsf{Stab}_t(g') \geq \mathsf{Stab}_t(f') - \epsilon$. Note that the main difference between the desired conclusion of Theorem 3 and what is accomplished in this step is that the ambient dimension remains n as opposed to a bounded dimension n_0.

Why is this true? The basic intuition is that if f is noise stable then it should have most of its Hermite expansion weight at low degree. Therefore we should be able to replace f with the PTF where the polynomial is the truncated expansion of f. There are a number of challenges in formalizing this intuition:

1. We cannot rule out that a positive fraction of the weight of f is at high degrees (perhaps as large as n);
2. it is not clear that the PTF obtained this way is noise stable; nor that
3. it has the right expected value.

Our analysis proceeds as follows. We would like to construct g' from f by "rounding" $P_t f$ for some small t. The advantage of $P_t f$ over f is that $P_t f$ is guaranteed to have decaying tails. The rounding of $P_t f$ can be performed given some $a \in \mathbb{R}^n$ by considering the function $g_a : \mathbb{R}^n \to [k]$ which takes the value i whenever i is the largest coordinate of $P_t f - a$. It is not hard to prove that it is possible to choose a such that $\mathbb{E}[g_a] = \mathbb{E}[f]$; moreover, one can show that this function g_a has better noise stability than f does. The main obstacle is that the function g_a is not a PTF. Unfortunately the Hermite decay of $P_t f$ does not translate to Hermite decay of g_a. Instead we use smoothed analysis to show that for most a's, g_a has Hermite decay and can therefore be well approximated by PTF. The smoothed analysis argument uses the co-area formula and gradient bounds and draws on ideas from [17, 14].

2.2 From PTF in dimension n to a small PTF of bounded degree polynomials

Given the function $g' : \mathbb{R}^n \to [k]$ of degree $d = d(t, k, \epsilon)$, our next goal is show it is possible to obtain a PTF g on some $n_0 = n_0(t, k, \epsilon)$ variables such that (i) $\|\mathbf{E}[g] - \mathbf{E}[g']\|_1 \leq \epsilon$ and (ii) $\mathrm{Z}|\langle g, P_t g\rangle - \langle g', P_t g'\rangle| \leq \epsilon$. This part builds on and extends the theory and results of [5]. The key notion introduced in [5] is that of an *eigenregular* polynomial. Namely, a polynomial is said to be δ-eigenregular if for the canonical tensor $\mathcal{A}_p$ associated with the polynomial, the ratio of the maximum singular value to its Frobenius norm is at most δ (the tensor notions are explicitly defined later).

The key advantage of this definition is that as shown in [5], when $\delta \to 0$, the distribution of p (under γ_n) converges to a normal. In other words, eigenregular polynomials obey a

central limit theorem. In fact, given k polynomials $p_1, \ldots, p_k$ which are δ-eigenregular, they also obey a multidimensional central limit theorem.

The *regularity lemma* from [5] implies that the polynomials $p_1, \ldots, p_k$ can be jointly expressed as bounded (in terms of t, k and ϵ) size polynomials in eigenregular homogenous polynomials $\{\mathrm{In}(p_{s,q,\ell})\}$ where $1 \le s \le k$, $1 \le q \le d$ and $1 \le \ell \le \mathrm{num}(s,q)$. In other words, we may write $p_s = \mathrm{Out}(p_s)(\{\mathrm{In}(p_{s,q,\ell})\}_{q\in[d],\ell\in[\mathrm{num}(s,q)]})$, where $\mathrm{num}(s) = \sum_{q=1}^{d} \mathrm{num}(s,q)$ is bounded in terms of t, k and ϵ and all the Inner polynomials are δ-eigenregular.

[5] used the statement above to conclude that the joint distribution of $p_1, \ldots, p_k$ can be approximated in a bounded dimension as we can replace each of the inner polynomials by a one dimensional Gaussian. For our application things are more delicate, as we are not only interested in the joint distribution of $p_1, \ldots, p_k$ but also in the noise stability of $p_1, \ldots, p_k$. For this reason it is important for us to maintain the degrees of the inner polynomials (each of which is homogenous) and not replace them with Gaussians.

2.3 A small PTF representation

In the final step of the proof, we maintain $\mathrm{Out}(p_s)$ and show how that polynomials $\{\mathrm{In}(p_{s,q,\ell})\}$ can be replaced by a collection of polynomials $\{\mathrm{In}(r_{s,q,\ell})\}$ in bounded dimensions (in t, k and ϵ) thus completing the proof. The fact that a collection of homogenous polynomials can be replaced by polynomials in bounded dimensions is a tensor analogue of the fact that for any k vectors in $\mathbb{R}^n$, there exist k vectors in $\mathbb{R}^k$ with the same matrix of inner products. Once such polynomials are found, it is not hard to construct eigenregular polynomials from them by averaging the polynomials over independent copies of random variables.

3 Applications

Given the wide applicability of Borell's isoperimetric result to combinatorics and theoretical computer science, we believe that Theorem 3 will also be widely applicable. We will now point out some applications of this theorem. First, by combining Theorem 3 with the invariance principle [16], we derive a weak k-ary analogue of "Majority is Stablest". The analogue of the Ornstein-Uhlenbeck operator is the so-called *Bonami-Beckner operator* defined as follows: for $\rho \in [-1,1]$ and $x \in \{-1,1\}^n$, let $\mathcal{D}_\rho(x)$ be the product distribution over $\{-1,1\}^n$ such that for $y \sim \mathcal{D}_\rho(x)$, for all $1 \le i \le n$, $\mathbf{E}[x_i y_i] = \rho$. Then, for any $f : \{-1,1\}^n \to \mathbb{R}^k$, $T_\rho f(x) = \mathbf{E}_{y\sim D_\rho(x)}[f(y)]$. Likewise, for any $i \in [n]$ and $z \in \{-1,1\}^{n-1}$, let $f_{z,-i} : \{-1,1\} \to \mathbb{R}^k$ denote the function obtained by restricting all but the i^{th} coordinate to z. Then, $\mathsf{Var}(f_{z,-i}) = \mathbf{E}_z[\|f_{z,-i} - \mathbf{E}_z[f_{z,-i}]\|_2^2]$. Define the influence of the i^{th} coordinate on f by

$$\mathsf{Inf}_i(f) = \mathbf{E}_{z\in\{-1,1\}^{n-1}}[\mathsf{Var}(f_{z,-i})].$$

▶ **Theorem 4.** *Given any $k \in \mathbb{N}$ and $\rho \in [0,1], \epsilon > 0$, $\exists n_0 = n_0(k,\epsilon,\rho), C = C(k,\epsilon,\rho)$ (which is explicitly computable) such that the following holds: For any $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_k) \in \Delta_k$ and $n \ge n_0$, there is an explicitly computable $g = g_{\boldsymbol{\mu},\rho,\epsilon} : \{-1,1\}^n \to [k]$ such that $\max_i \mathsf{Inf}_i(g) \le C/\sqrt{n}$ and $|\Pr_{x\in\{-1,1\}^{n_0}}[g(x) = i] - \mu_i| \le C/\sqrt{n}$ and for any $f : \{-1,1\}^n \to [k]$ such that $|\Pr_{x\in\{-1,1\}^n}[f(x) = i] - \mu_i| \le \epsilon$ and $\max_i \mathsf{Inf}_i(f) \le \epsilon$,*

$$\mathbf{E}[\langle f, T_\rho f\rangle] \le \mathbf{E}[\langle g, T_\rho g\rangle] + 2k\epsilon.$$

We give a brief sketch of the proof, which is a straightforward consequence of the invariance principle in one direction and the central limit theorem in the other. First, the invariance

principle implies that the discrete noise stability cannot be much larger than the best Gaussian noise stability. It remains, then, to construct a boolean function g whose noise stability is almost the same as the best possible Gaussian noise stability. For this, suppose that $n = n_1 n_2$ (for some n_1 and n_2 that will need to be sufficiently large), and let $h : \mathbb{R}^{n_1} \to [k]$ have almost optimal Gaussian noise stability. Then define the boolean function $g(x) = h(z(x))$, where $z \in \mathbb{R}^{n_1}$ has ith coordinate $n_2^{-1/2} \sum_{j=in_2+1}^{(i+1)n_2} x_j$. When n_2 is sufficiently large, the central limit theorem implies that the boolean noise stability of g is approximately the Gaussian noise stability of h, and so it is almost optimal.

The same result holds for other domains, for example for $f, g : [k]^n \to [k]$. In particular, the case where $(\mu_1, \ldots, \mu_k) = (1/k, \ldots, 1/k)$ implies that in a tied elections between k alternatives, we can find an ϵ-optimally robust noise stable voting rule, where the stability is with respect of each candidate randomizing their vote independently with probability $1 - \rho$.

3.1 Relationship to rounding of SDPs

To the best of our knowledge our results are independent of the the results of Raghavendra and Steurer [18] who showed that for any CSP, there is an a rounding algorithm that is optimal up to ϵ, whose running time is polynomial in the instance size and doubly exponential in $1/\epsilon$. It is natural to suspect that the two results are related, since there are well-known connections between SDP rounding and Gaussian noise stability.

However, the usual analysis relating rounding to Gaussian noise stability seems to require that halfspaces maximize noise stability for all possible values of the noise. In our results, this property does not necessarily hold.

In the other direction, it is tempting to try to cast the noise stability problem as an optimization problem on Gaussian graphs and then apply the results of Steurer and Raghavendra to obtain explicit bounds on the dimension where an almost optimal solution can be achieved. It is hard to implement this approach for two reasons: first, we do not know the SDP solution for the Gaussian graph; second, we are interested in the optimal solution and it is not clear what is the relation between the best integral solution and the SDP solution for the Gaussian graph.

While we do not see how to formally relate the two works, connecting the two (if possible) would surely yield important insights.

3.2 Non-interactive correlation distillation

Next, we talk about a basic problem in information theory and communication complexity which was recently considered in the work of Ghazi, Kamath and Sudan [6]. Let there be two non-communicating players Alice and Bob who have access to independent samples from a joint distribution $\mathbf{P} = (\mathbf{X}, \mathbf{Y})$ on the set $\mathcal{A} \times \mathcal{B}$. In other words, Alice (resp. Bob) have access to $(x_1, x_2, \ldots,)$ (resp. $(y_1, y_2, \ldots)$) such that $x_i \in \mathcal{A}$, $y_i \in \mathcal{B}$ and for each $i \in \mathbb{N}$, (x_i, y_i) is distributed according to $\mathbf{P}$, and the random variables $\{(x_i, y_i)\}_{i=1}^n$ are mutually independent. Let $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_k) \in \Delta_k$ and $\boldsymbol{\nu} = (\nu_1, \ldots, \nu_k) \in \Delta_k$. What is the maximum $\kappa \in [0, 1]$ such that Alice and Bob can non-interactively jointly sample a distribution $\mathbf{Q}$ on $[k] \times [k]$ such that the distribution of the marginals of Alice and Bob are $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ respectively and they sample the same output with probability κ?

To formulate this problem more precisely, we introduce some notation. Let $\mathbf{X}^n = (\mathbf{X}_1, \ldots, \mathbf{X}_n)$ and $\mathbf{Y}^n = (\mathbf{Y}_1, \ldots, \mathbf{Y}_n)$, where $(\mathbf{X}_i, \mathbf{Y}_i)$ are independently drawn from $\mathbf{P}$. Now, note that a non-interactive protocol for Alice and Bob is equivalent to a pair (f, g) where $f : \mathcal{A}^n \to [k]$ and $g : \mathcal{B}^n \to [k]$ (for some $n \in \mathbb{N}$). In this terminology, the question

now becomes the following: given $\boldsymbol{\mu}$, $\boldsymbol{\nu}$, do there exist n and $f : \mathcal{A}^n \to [k]$ and $g : \mathcal{B}^n \to [k]$ such that $f(\mathbf{X}^n) \sim \boldsymbol{\mu}$, $g(\mathbf{Y}^n) \sim \boldsymbol{\nu}$, and $\Pr(f(\mathbf{X}^n) = g(\mathbf{Y}^n)) = \kappa$?

Before we state the main result of [6] and our extension, we consider a motivating example. Let $\mathcal{A} = \mathcal{B} = \mathbb{R}$ and let $\mathbf{P} = (\mathbf{X}, \mathbf{Y})$ be two ρ-correlated standard Gaussians. Let $k = 2$ and $\boldsymbol{\mu} = \boldsymbol{\nu}$. Then, Borell's isoperimetric theorem (Theorem 1) states that the maximum achievable κ is given by

$$\kappa = \Pr_{\mathbf{X},\mathbf{Y}}[f(\mathbf{X}) = f(\mathbf{Y})],$$

where $f : \mathbb{R} \to \Delta_2$ is a halfspace with measure $\boldsymbol{\mu}$. Thus, in the above case, $n = 1$ suffices and $f = g$ is the halfspace whose measure is $\boldsymbol{\mu}$. We now state the main result of [6]. For the result below, for probability distribution $\mathbf{P}$, we let $|\mathbf{P}|$ denote the size of some standard encoding of $\mathbf{P}$.

▶ **Theorem 5.** *[Ghazi-Kamath-Sudan] Let $(\mathcal{A} \times \mathcal{B}, \mathbf{P})$ be a probability space, and let $\mathbf{X}^n$ and $\mathbf{Y}^n$ be as above. There is an algorithm running in time $O_{|\mathbf{P}|,\delta}(1)$ such that given $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ in Δ_2 and a parameter $\kappa \in [0,1]$, it distinguishes between the following two cases:*

1. *There exist $n \in \mathbb{N}$, $f : \mathcal{A}^n \to \{0,1\}$ and $g : \mathcal{B}^n \to \{0,1\}$ such that $f(\mathbf{X}^n) \sim \boldsymbol{\mu}$, $g(\mathbf{Y}^n) \sim \boldsymbol{\nu}$, and $\Pr(f(\mathbf{X}^n) = g(\mathbf{Y}^n)) \geq \kappa - \delta$. In this case, there is an explicit $n_0 = n_0(|\mathbf{P}|, \delta)$ such that we may choose $n \leq n_0$. Further, the functions f and g are explicitly computable.*
2. *For any $n \in \mathbb{N}$ and $f : \mathcal{A}^n \to \{0,1\}$ and $g : \mathcal{B}^n \to \{0,1\}$, if $g : \mathcal{B}^n \to \{0,1\}$ satisfy $f(\mathbf{X}^n) \sim \boldsymbol{\mu}$ and $g(\mathbf{Y}^n) \sim \boldsymbol{\nu}$ then $\Pr(f(\mathbf{X}^n) = g(\mathbf{Y}^n)) \leq \kappa - 8\delta$.*

In other words, the above theorem states that there is an algorithm which, given $\mathbf{P}$, target marginals $\boldsymbol{\mu}$, $\boldsymbol{\nu}$, and correlation κ, can distinguish between two cases: (a) In the first case, it is possible for Alice and Bob to non-interactively simulate a distribution which has the correct marginals and achieves the correlation κ up to δ. In this case, there is an explicit bound on the number of copies of $\mathbf{P}$ required and the algorithm also outputs the functions f, g used for the non-interactive simulation. (b) In the second case, no non-interactive protocol between Alice and Bob can simulate a distribution that has the correct marginals and target correlation to error at most 8δ.

We remark that the requirement for the marginals to match *exactly* is unimportant. Indeed, any protocol where the marginals match approximatly can be "fixed" to have exact marginals, with a small loss in the correlation.

The main restriction of Theorem 5 is that the output of the non-interactive protocol is a pair of bits. Theorem 3 immediately implies the following modification of Theorem 5: we may replace $\{0,1\}$ by $[k]$ wherever it appears, provided that we also assume that $\mathbf{P}$ is a Gaussian measure. In the best of both worlds, we would be able to replace $\{0,1\}$ by $[k]$ without adding the assumption that $\mathbf{P}$ is a Gaussian measure. Using the methods we develop here, this also turns out to be possible; we provide details in a follow-up work.

4 Reduction from arbitrary functions to PTFs

In this section, we will prove that given any k-ary function with a given set of measures for each of the k-partitions, there is a multivariate PTF with nearly the same measures for the induced partitions which is a multivariate PTF and (up to an error ϵ), no less noise stable at a fixed noise rate t. This is the first step ("from general partitions to PTF") of the proof sketch in Section 2.

For technical reasons, we will also require the PTFs to satisfy a property which we refer to as (d, δ)-*balanced* defined below.

▶ **Definition 6.** A degree-d multivariate PTF $f = \mathsf{PTF}(p^{(1)}, \ldots, p^{(k)})$ is said to be (d, δ)-balanced if each $p^{(i)}$ has variance 1 and $|\mathbf{E}[p^{(i)}]| \leq \log^{d/2}(k \cdot d/\delta)$.

We ask the reader to observe that the first condition (namely, $\mathsf{Var}(p^{(i)}) = 1$) can be achieved without loss of generality by simply scaling all the polynomials to have variance 1. This scaling does not change the value of the PTF at any point x. While the condition on expectation is non-trivial, the next proposition says that any multivariate PTF can be assumed to be (d, δ)-balanced while only changing the value of the PTF at δ-fraction of places.

▶ **Lemma 7.** *Let $f : \mathbb{R}^n \to [k]$ defined as $f = \mathsf{PTF}(p^{(1)}, \ldots, p^{(k)})$. Then, there is a (d, δ)-balanced multivariate PTF $g : \mathbb{R}^n \to [k]$ defined as $g = \mathsf{PTF}(q^{(1)}, \ldots, q^{(k)})$ such that $\Pr_x[g(x) \neq f(x)] \leq \delta$. Further, $\Pr_x[x \in \mathsf{Collision}(g)] \leq \Pr_x[x \in \mathsf{Collision}(f)] + \delta$. In fact, the polynomials $\{q^{(i)}\}$ are linear translations of the polynomials $\{p^{(i)}\}$.*

The proof of the lemma is deferred to the full version of the paper. The next theorem is the main result of this section namely, that given any function $f : \mathbb{R}^n \to [k]$, it is possible to obtain a multivariate (d, ϵ) balanced PTF g (for some explicit $d = O_{\epsilon,k}(1)$) such that the resulting PTF g has nearly the same partition sizes and noise stability. Further, $\mathsf{Collision}(g)$ has small probability.

▶ **Theorem 8.** *Let $f : \mathbb{R}^n \to [k]$ satisfy $\mathbb{E}[f] = \boldsymbol{\mu}$ where $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_k)$. Then, for every $\epsilon > 0$, there exists a multivariate PTF $g : \mathbb{R}^n \to [k]$ of degree $d = d(k, \epsilon)$ such that*

- $\|\mathbf{E}[g] - \boldsymbol{\mu}\|_1 \leq \epsilon$,
- $\langle g, P_t g \rangle \geq \langle f, P_t f \rangle - \epsilon$,
- $\Pr[x \in \mathsf{Collision}(g)] \leq \epsilon$, *and*
- *g is (d, ϵ)-balanced.*

We will prove Theorem 8 in two parts. The first step is to show that we can replace f by a function with explicit Hermite decay:

▶ **Lemma 9.** *For every $f : \mathbb{R}^n \to [k]$ and every $\epsilon > 0$, there exists a function $h : \mathbb{R}^n \to [k]$ satisfying the following:*

- $\mathbb{E}[h] = \mathbb{E}[f]$,
- $\langle h, P_t h \rangle \geq \langle f, P_t f \rangle - \epsilon$,
- $\mathsf{W}^{>d}[h] \leq \epsilon$ *for some explicit $d = d(k, \epsilon)$.*

The second step in the proof of Theorem 8 goes from explicit Hermite decay to an actual PTF:

▶ **Lemma 10.** *Let $h : \mathbb{R}^n \to [k]$ such that $\mathsf{W}^{>d}[h] \leq \epsilon$. Then, there is a PTF $g : \mathbb{R}^n \to [k]$ of degree d such that $\mathbf{E}_x[g(x) \neq h(x)] \leq k^2 \cdot \epsilon$. Further, $\Pr_x[x \in \mathsf{Collision}(h)] \leq k^2 \cdot \epsilon$.*

It is easy to see that Theorem 8 follows in a straightforward way by combining Lemma 9 and Lemma 10. Note that the condition of g being (d, ϵ)-balanced is obtained by simply applying Proposition 7.

While we defer the proof of both Lemma 9 and Lemma 10 to the full version, as the proof of Lemma 9 is more non-standard, we give a brief overview here. The first observation is that bounding $\mathbb{E}[|\nabla h|]$ implies a bound on $\mathsf{W}^{>d}[h]$. This follows a standard spectral argument stated below.

▶ **Lemma 11.** *Let $f : \mathbb{R}^n \to [0, 1]$ be of bounded variation. Then,*

$$\mathsf{W}^{\geq d}[f] \leq O\left(\frac{1}{\sqrt{d}}\right) \cdot \mathbf{E}[|\nabla f|].$$

The second observation is that the function h obtained by thresholding $P_t f$ at a suitable value (chosen, for example, so that $\mathbb{E}[h] = \mathbb{E}[f]$) satisfies $\langle h, P_t h\rangle \geq \langle f, P_t f\rangle$. This is stated below.

▶ **Lemma 12.** *Let $f : \mathbb{R}^n \to \{e_1, \ldots, e_k\}$ and take $t > 0$. If $g \in T_z(f)$ for some $z \in \mathbb{R}^k$ satisfies $\mathbb{E}[g] = \mathbb{E}[f]$ then $\mathsf{Stab}_t(g) \geq \mathsf{Stab}_t(f)$.*

Based on the previous paragraph, it seems like we would like to bound $\mathbb{E}[|\nabla h|]$ where h is obtained by thresholding $P_t f$; let $a \in \mathbb{R}^k$ be the desired threshold value, so that thresholding $P_t f$ at a produces a partition with the right measures. It turns out, unfortunately, that for h defined in this way, $\mathbb{E}[|\nabla h|]$ could be arbitrarily large. A key insight of [14] is that (using the co-area formula and gradient bounds on $P_t f$) the partition produced by thresholding at a random value near a has bounded expected surface area. In particular, although thresholding exactly at a might be a bad idea, there exist many good nearby values at which to threshold. Based on this observation, we construct h in two steps. In the first step, we define h by thresholding $P_t f$, but only on the set of $x \in \mathbb{R}^n$ for which $P_t f(x)$ is not too close to a. By choosing "not too close" in a suitable random way, the observation of [14] implies that this step only contributes a bounded amount to $\mathbb{E}[|\nabla h|]$. Since the first step is almost the same as just thresholding $P_t f$ at a, it is consistent with our desire that $\langle h, P_t h\rangle \geq \langle f, P_t f\rangle - \epsilon$.

In the second step, we partition the remaining part of $\mathbb{R}^n$ by chopping it with halfspaces of the correct size. Since halfspaces have a bounded surface area, this also contributes a bounded amount to $\mathbb{E}[|\nabla h|]$. Crucially, this step does not destroy the value of $\langle h, P_t h\rangle$; fundamentally, this is because $P_t f$ is almost constant on the set we are partitioning. The actual details of the proof of Lemma 10 can be found in the full version. Let us now move to the second major technical ingredient of this paper.

5 Reduction from PTFs to PTFs on a constant number of variables

The second big technical ingredient required in the paper is the following:

▶ **Theorem 13.** *Let $f : \mathbb{R}^n \to [k]$ be a degree-d, (d, ϵ)-balanced PTF with $\mathbb{E}_x[f(x)] = \boldsymbol{\mu}$ where $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_k)$. Further, let us assume that $\Pr[x \in \mathsf{Collision}(f)] \leq \epsilon/(40k^2)$. Then, for every $\epsilon > 0$, there exists a degree-d PTF $f_{\mathsf{junta}} : \mathbb{R}^{n_0} \to [k]$ such that:*

- $\|\mathbf{E}_x[f_{\mathsf{junta}}(x)] - \boldsymbol{\mu}\|_1 \leq \epsilon$,
- $\langle f_{\mathsf{junta}}, P_t f_{\mathsf{junta}}\rangle \geq \langle f, P_t f\rangle - \epsilon$.

Further, $n_0 = n_0(d, k, \epsilon)$ is an explicitly defined function.

The above theorem states that given a degree-d multivariate PTF over n variables, there is another multivariate PTF which induces approximately the same partition sizes and has approximately the same noise stability (at any fixed noise rate t) but the new PTF is only over some (explicitly defined) $O_{d,t}(1)$ variables. The main workhorse for proving this thoerem are two structural theorems for low-degree polynomials proven in [5]. Unfortunately, even stating these theorems require significantly cumbersome technical definitions. In particular, we will need to define the relation between polynomials and tensors and then define the notion of an ϵ-*eigenregular* polynomial. We avoid doing that in this version of the paper and refer the reader to the full version of the paper. However, we observe that Theorem 3 follows very easily by combining Theorem 13 and Theorem 8.

Proof of Theorem 3

Let us us assume that given measure $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_k)$ and noise rate $t > 0$, the most noise stable partition is $f : \mathbb{R}^n \to [k]$. Then, applying Theorem 8 (with error parameter

$\epsilon/(40k^2)$), we obtain a PTF g_1 of degree $d = O_{k,\epsilon}(1)$ such that it is $(d, \epsilon/2)$ balanced and $\Pr[x \in \mathsf{Collision}(g_1)] \leq \epsilon/(40k^2)$. Further, note that $\|\mathbf{E}[g_1] - \mathbf{E}[f]\|_1 \leq \epsilon/2$ and $\langle g_1, P_t g_1 \rangle \geq \langle f, P_t f \rangle - \epsilon/2$.

We next apply Theorem 13 on this function g_1 to obtain f_{junta} which is a degree-d PTF on $n_0 = O_{d,\epsilon,k}(1)$ variables such that $\|\mathbf{E}[g_1] - \mathbf{E}[f_{\mathsf{junta}}]\|_1 \leq \epsilon/2$ and $\langle f_{\mathsf{junta}}, P_t f_{\mathsf{junta}} \rangle \geq \langle g_1, P_t g_1 \rangle - \epsilon/2$.

Combining these two facts, we obtain $\|\mathbf{E}[g_1] - \mathbf{E}[f_{\mathsf{junta}}]\|_1 \leq \epsilon$ and $\langle f_{\mathsf{junta}}, P_t f_{\mathsf{junta}} \rangle \geq \langle f, P_t f \rangle - \epsilon$. Setting $g = f_{\mathsf{junta}}$ concludes the proof.

Acknowledgments. The authors started this research at the workshop on "Information theory and concentration phenomena" at the IMA, Minneapolis in 2015. The authors would like to thank the organizers of this workshop and the IMA for their hospitality.

References

1 S. G. Bobkov. Isoperimetric problem for uniform enlargement. *Studia Math*, 123(1):81–95, 1997.

2 C. Borell. The Brunn-Minkowski inequality in Gauss space. *Invent. Math.*, 30:207–216, 1975.

3 C. Borell. Geometric bounds on the Ornstein-Uhlenbeck velocity process. *Z. Wahrsch. Verw. Gebiete*, 70(1):1–13, 1985.

4 J. Corneli, I. Corwin, S. Hurder, V. Sesum, Y. Xu, E. Adams, D. Davis, M. Lee, R. Visocchi, N. Hoffman, and Robert M. Hardt. Double bubbles in gauss space and spheres. *Houston Journal of Mathematics*, 34(1):181–204, 2008.

5 A. De and R. Servedio. Efficient deterministic approximate counting for low-degree polynomial threshold functions. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA*, pages 832–841, 2014. Full version at http://arxiv.org/abs/1311.7178.

6 B. Ghazi, P. Kamath, and M. Sudan. Decidability of non-interactive simulation of joint distributions. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 545–554, 2016.

7 S. Heilman, E. Mossel, and J. Neeman. Standard simplices and pluralities are not the most noise stable. *Israel Journal of Mathematics*, pages 1–21, 2014.

8 Steven Heilman. Euclidean partitions optimizing noise stability. *Electron. J. Probab*, 19(71):1–37, 2014.

9 M. Hutchings, F. Morgan, M. Ritoré, and A. Ros. Proof of the double bubble conjecture. *Res. Announc. Amer. Math. Soc*, 6:45–49, 2000.

10 M. Hutchings, F. Morgan, M. Ritoré, and A. Ros. Proof of the double bubble conjecture. *Ann. of Math.*, 155:459–489, 2002.

11 G. Kalai. A Fourier-theoretic perspective on the Concordet paradox and Arrow's theorem. *Adv. in Appl. Math.*, 29(3):412–426, 2002.

12 S. Khot, G. Kindler, E. Mossel, and R. O'Donnell. Optimal inapproximability results for Max-Cut and other 2-variable CSPs? In *Proc. 45th Annual IEEE Symposium on Foundations of Computer Science*, pages 146–154, 2004.

13 S. Khot, G. Kindler, E. Mossel, and R. O'Donnell. Optimal inapproximability results for max-cut and other 2-variable csps? *SIAM J. Comput.*, 37:319–357, 2007. URL: `http://www.stat.berkeley.edu/~mossel/publications/max_cut_final.pdf`.

14 P. Kothari, A. Nayyeri, R. O'Donnell, and C. Wu. Testing surface area. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 1204–1214, 2014.

15 M. Ledoux. Semigroup proofs of the isoperimetric inequality in Euclidean and Gauss space. *Bull. Sci. Math.*, 118:485–510, 1994.

16 E. Mossel, R. O'Donnell, and K. Oleszkiewicz. Noise stability of functions with low influences: Invariance and optimality. *Ann. Math.*, 171(1):295–341, 2010.

17 J. Neeman. Testing surface area with arbitrary accuracy. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 393–397, 2014.

18 Prasad Raghavendra and David Steurer. How to round any csp. In *Foundations of Computer Science, 2009. FOCS'09. 50th Annual IEEE Symposium on*, pages 586–594. IEEE, 2009.

19 Ben W. Reichardt, Cory Heilmann, Yuan Y. Lai, and Anita Spielman. Proof of the double bubble conjecture in r4 and certain higher dimensional cases. *Pacific Journal of Mathematics*, 208(2):347–366, 2003.

20 V. Sudakov and B. Tsirel'son. Extremal properties of half-spaces for spherically invariant measures. *J. Soviet Math.*, 9:9–18, 1978. Translated from Zap. Nauchn. Sem. Leningrad. Otdel. Math. Inst. Steklova. 41 (1974), 14–21.

From Weak to Strong LP Gaps for All CSPs*

Mrinalkanti Ghosh[1] **and Madhur Tulsiani**[2]

1 **Toyota Technological Institute at Chicago, Chicago, IL, USA**
mkghosh@ttic.edu
2 **Toyota Technological Institute at Chicago, Chicago, IL, USA**
madhurt@ttic.edu

Abstract

We study the approximability of constraint satisfaction problems (CSPs) by linear programming (LP) relaxations. We show that for every CSP, the approximation obtained by a basic LP relaxation, is no weaker than the approximation obtained using relaxations given by $\Omega\left(\frac{\log n}{\log\log n}\right)$ levels of the Sherali-Adams hierarchy on instances of size n.

It was proved by Chan et al. [FOCS 2013] (and recently strengthened by Kothari et al. [STOC 2017]) that for CSPs, any polynomial size LP extended formulation is no stronger than relaxations obtained by a super-constant levels of the Sherali-Adams hierarchy. Combining this with our result also implies that any polynomial size LP extended formulation is no stronger than simply the *basic* LP, which can be thought of as the base level of the Sherali-Adams hierarchy. This essentially gives a dichotomy result for approximation of CSPs by polynomial size LP extended formulations.

Using our techniques, we also simplify and strengthen the result by Khot et al. [STOC 2014] on (strong) approximation resistance for LPs. They provided a necessary and sufficient condition under which $\Omega(\log\log n)$ levels of the Sherali-Adams hierarchy cannot achieve an approximation better than a random assignment. We simplify their proof and strengthen the bound to $\Omega\left(\frac{\log n}{\log\log n}\right)$ levels.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Constraint Satisfaction Problem, Convex Programming, Linear Programming Hierarchy, Integrality Gap

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.11

1 Introduction

Given a finite alphabet $[q] = \{0, \ldots, q-1\}$ and a predicate $f : [q]^k \to \{0, 1\}$, an instance of the problem MAX k-CSP(f) consists of (say) m constraints over a set of n variables $x_1, \ldots, x_n$ taking values in the set $[q]$. Each constraint C_i is of the form $f(x_{i_1} + b_{i_1}, \ldots, x_{i_k} + b_{i_k})$ for some k-tuple of variables $(x_{i_1}, \ldots x_{i_k})$ and $b_{i_1}, \ldots, b_{i_q} \in [q]$, and the addition is taken to be modulo q. We say an assignment σ to the variables satisfying the constraint C_i if $C_i(\sigma(x_{i_1}), \ldots, \sigma(x_{i_k})) = 1$. Given an instance Φ of the problem, the goal is to find an assignment σ to the variables satisfying as many constraints as possible. The approximability of the MAX k-CSP(f) problem has been extensively studied for various predicates f (see e.g., [29] for a survey), and special cases include several interesting and natural problems such as MAX 3-SAT, MAX 3-XOR and MAX-CUT.

* This research was supported by supported by the National Science Foundation under award number CCF-1254044.

COMPUTATIONAL COMPLEXITY CONFERENCE

A topic of much recent interest has been the efficacy of Linear Programming (LP) and Semidefinite Programming (SDP) relaxations. For a given instance Φ of MAX k-CSP(f), let OPT(Φ) denote the *fraction* of constraints satisfied by an optimal assignment, and let FRAC(Φ) denote the value of the convex (LP/SDP) relaxation for the problem. Then, the performance guarantee of this algorithm is given by the integrality gap which equals the supremum of $\frac{\mathsf{FRAC}(\Phi)}{\mathsf{OPT}(\Phi)}$, over all instances Φ.

The study of unconditional lower bounds for general families of LP relaxations was initiated by Arora, Bollobás and Lovász [2] (see also [3]). They studied the Lovász-Schrijver [24] LP hierarchy and proved lower bounds on the integrality gap for Minimum Vertex Cover (their technique also yields similar bounds for MAX-CUT). De la Vega and Kenyon-Mathieu [13] and Charikar, Makarychev and Makarychev [12] proved a lower bound of $2 - o(1)$ for the integrality gap of the LP relaxations for MAX-CUT given respectively by $\Omega(\log \log n)$ and $n^{\Omega(1)}$ levels of the Sherali-Adams LP hierarchy [28]. Several follow-up works have also shown lower bounds for various other special cases of the MAX k-CSP problem, both for LP and SDP hierarchies [1, 27, 32, 26, 7, 5, 21].

A recent result by Chan et al. [8] shows a connection between strong lower bounds for the Sherali-Adams hierarchy, and lower bounds on the size of LP extended formulations for the corresponding problem. In fact, their result proved a connection not only for a lower bound on the worst case integrality gap, but for the entire *approximability curve*. We say that Φ is (c, s)-integrality gap instance for a relaxation of MAX k-CSP(f), if we have FRAC(Φ) $\geq c$ and OPT(Φ) $< s$. And we say that Φ is (c, s)-approximable by a relaxation of MAX k-CSP(f), if for instances with OPT(ϕ) $< s$, we have FRAC(Φ) $\leq c$. They showed that for any fixed $t \in \mathbb{N}$, if there exist (c, s)-integrality gap instances of size n for the relaxation given by t levels of the Sherali-Adams hierarchy, then for all $\varepsilon > 0$ and sufficiently large N, there exists a $(c - \varepsilon, s + \varepsilon)$ integrality gap instance of size (number of variables) N, for any linear extended formulation of size at most $N^{t/2}$. They also give a tradeoff when t is a function of n. This was recently improved by Kothari et al. [20] and we describe the improved tradeoff later.

We strengthen the above results by showing that for all $c, s \in [0, 1]$, (c, s)-integrality gap instances for a "basic LP" can be used to construct $(c - \varepsilon, s + \varepsilon)$ integrality gap instances for $\Omega_\varepsilon\left(\frac{\log n}{\log \log n}\right)$ levels of the Sherali-Adams hierarchy. The basic LP uses only a subset of the constraints in the relaxation given by k levels of the Sherali-Adams hierarchy for MAX k-CSP(f). In particular, this shows that a lower bound on the integrality gap for even the basic LP, implies a similar lower bound on the integrality gap of any polynomial size extended formulation. This can also be viewed as a dichotomy result showing that for any predicate f, either MAX k-CSP(f) is (c, s)-approximable by the *basic LP relaxation* (which is of size linear in the size of the instance) or for all $\varepsilon > 0$, a $(c - \varepsilon, s + \varepsilon)$ cannot be achieved by *any polynomial sized LP extended formulation.* We note that both the above results and our result apply to all f, q and all $c, s \in [0, 1]$.

1.1 Comparison with (implications of) Raghavendra's UGC hardness result

A remarkable result by Raghavendra [25] shows that a (c, s)-integrality gap instance for a "basic SDP" relaxation of MAX k-CSP(f) implies hardness of distinguishing instances Φ with OPT(Φ) $< s$ from instances with OPT(Φ) $\geq c$, assuming the Unique Games Conjecture (UGC) of Khot [15]. The basic SDP considered by Raghavendra involves moments for all pairs of variables, and all subsets of variables included in a constraint. The basic LP we consider is weaker than this SDP and does not contain the positive semidefiniteness constraint.

Combining Raghavendra's result with known constructions of integrality gaps for Unique Games by Raghavendra and Steurer [26], and by Khot and Saket [16], one can obtain a result qualitatively similar to ours, for the mixed hierarchy. In particular, a (c, s) integrality gap for the basic SDP implies a $(c-\varepsilon, s+\varepsilon)$ integrality gap for $\Omega((\log\log n)^{1/4})$ levels of the mixed hierarchy.

Note however, that the above result is incomparable to our result, since it starts with stronger hypothesis (a basic SDP gap) and yields a gap for the mixed hierarchy as opposed to the Sherali-Adams hierarchy. While the above can also be used to derive lower bounds for linear extended formulations, one needs to start with an SDP gap instance to derive an LP lower bound. The basic SDP is known to be provably stronger than the basic LP for several problems including various 2-CSPs. Also, for the worst case f for $q = 2$, the integrality gap of the basic SDP is $O(2^k/k)$ [11], while that of the basic LP is 2^{k-1}.

A recent result by Khot and Saket [17] shows a connection between the integrality gaps for the basic LP and those for the basic SDP. They prove that, assuming the UGC, a (c, s) integrality gap instance for the basic LP implies an NP-hardness of distinguishing instances Φ with $\mathsf{OPT}(\Phi) \geq \Omega\left(\frac{c}{k^3 \cdot \log(q)}\right)$ from instances with $\mathsf{OPT}(\Phi) \leq 4s$. Their result also shows that a (c, s) integrality gap instance for the basic LP can be used to produce a $\left(\Omega\left(\frac{c}{k^3 \cdot \log(q)}\right), 4s\right)$ integrality gap instance for the basic SDP, and hence for $\Omega((\log\log n)^{1/4})$ levels of the mixed hierarchy.

1.2 Other related work

The power of the basic LP for solving valued CSPs *to optimality* has been studied in several previous works. These works consider the problem of minimizing the penalty for unsatisfied constraints, where the penalties take values in $\mathbb{Q} \cup \{\infty\}$. Also, they study the problem not only in terms of single predicate f, but rather in terms of the constraint language generated by a given set of (valued) predicates.

It was shown by Thapper and Živný [30] that when the penalties are finite-valued, if the problem of finiding the optimum solution cannot be solved by the basic LP, then it is NP-hard. Kolmogorov, Thapper and Živný [19] give a characterization of CSPs where the problem of minimizing the penalty for unsatisfied constraints can be solved *exactly* by the basic LP. Also, a recent result by Thapper and Živný [31] shows the valued CSP problem for a constraint language can be solved to optimality by a bounded number of levels of the Sherali-Adams hierarchy if and only if it can be solved by a relaxation obtained by augmenting the basic LP with contraints implied by three levels of the Sherali-Adams hierarchy. However, the above works only consider the case when the LP gives an exact solution, and do not focus on approximation.

The techniques from [12] used in our result, were also extended by Lee [23] to prove a hardness for the Graph Pricing problem. Kenkre et al. [14] also applied these to show the optimality of a simple LP-based algorithm for Digraph Ordering.

1.3 Our results

Our main result is the following.

▶ **Theorem 1.** *Let $f : [q]^k \to \{0,1\}$ be any predicate. Let Φ_0 be a (c, s) integrality gap instance for basic LP relaxation of MAX k-CSP (f). Then for every $\varepsilon > 0$, there exists $c_\varepsilon > 0$ such that for infinitely many $N \in \mathbb{N}$, there exist $(c-\varepsilon, s+\varepsilon)$ integrality gap instances of size N for the LP relaxation given by $c_\varepsilon \cdot \frac{\log N}{\log\log N}$ levels of the Sherali-Adams hierarchy.*

Combining the above with the connection between Sherali-Adams gaps and extended formulations by [8, 20] yields the following corollary:

▶ **Corollary 2.** *Let $f : [q] \to \{0,1\}$ be any predicate. Let Φ_0 be a (c,s) integrality gap instance for basic LP relaxation of MAX k-CSP (f). Then for every $\varepsilon > 0$, there exists $c'_\varepsilon > 0$ such that for infinitely $N \in \mathbb{N}$, there exist $(c-\varepsilon, s+\varepsilon)$ integrality gap instances of size N, for every linear extended formulation of size at most $N^{c'_\varepsilon \cdot \frac{\log N}{\log\log N}}$.*

As an application of our methods, we also simplify and strengthen the approximation resistance results for LPs proved by Khot et al. [18]. They studied predicates $f : \{0,1\}^k \to \{0,1\}$ and provided a necessary and sufficient condition for the predicate to be strongly approximation resistant for the Sherali-Adams LP hierarchy. We say a predicate is strongly approximation resistant if for all $\varepsilon > 0$, it is hard to distinguish instances Φ for which $\left|\mathsf{OPT}(\Phi) - \mathbb{E}_{x\in\{0,1\}^k}[f(x)]\right| \leq \varepsilon$ from instances with $\mathsf{OPT}(\Phi) \geq 1-\varepsilon$. In the context of the Sherali-Adams hierarchy, they showed that when this condition is satisfied, there exist instances Φ satisfying $\left|\mathsf{OPT}(\Phi) - \mathbb{E}_{x\in\{0,1\}^k}[f(x)]\right| \leq \varepsilon$ and $\mathsf{FRAC}(\Phi) \geq 1-\varepsilon$, where $\mathsf{FRAC}(\Phi)$ is the value of the relaxation given by $O_\varepsilon(\log\log n)$ levels of the Sherali-Adams hierarchy. We strengthen their result (and provide a simpler proof) to prove the following.

▶ **Theorem 3.** *Let $f : \{0,1\}^k \to \{0,1\}$ be any predicate satisfying the condition for strong approximation resistance for LPs, given by [18]. Then for every $\varepsilon > 0$, there exists $c_\varepsilon > 0$ such that infinitely many $N \in \mathbb{N}$, there exists an instance Φ of MAX k-CSP(f) of size N, satisfying*

$$\left|\mathsf{OPT}(\Phi) - \mathop{\mathbb{E}}_{x\in\{0,1\}^k}[f(x)]\right| \;\leq\; \varepsilon \qquad \mathsf{FRAC}(\Phi) \;\geq\; 1-\varepsilon\,,$$

where $\mathsf{FRAC}(\Phi)$ is the value of the relaxation given by $c_\varepsilon \cdot \frac{\log N}{\log\log N}$ levels of the Sherali-Adams hierarchy.

As before, the above theorem also yields a corollary for extended formulations.

1.4 Proof overview and techniques

1.4.1 The gap instance

The construction of our gap instances is inspired by the construction by Khot et al. [18]. They gave a generic construction to prove integrality gaps for any approximation resistant predicate (starting from certificates of hardness in form of certain "vanishing measures"), and we use similar ideas to give a construction which can start from a basic LP integrality gap instance as a certificate, to produce a gap instance for a large number of levels. This construction is discussed in Section 5.

Given an integrality gap instance Φ_0 on n_0 variables, we treat this as a "template" (as in Raghavendra [25]) and generate a random instance using this. Concretely, we generate a new instance Φ on n_0 sets of n variables each. To generate a contraint, we sample a random constraint $C_0 \in \Phi_0$, and pick a variable randomly from each of the sets corresponding to variables in C_0. Thus, the instances generated are n_0-partite random hypergraphs, with each edge being generated according to a specified "type" (indices of sets to chose vertices from).

Note that previous instances of gap constructions for LP and SDP hierarchies were (hyper)graphs generated according to the model $\mathcal{G}_{n,p}$ with the signs of the literals chosen independently at random. However, proving an LP/SDP lower bound using such instances implies a strong result: it in fact proves that the predicate f is *useless* for the corresponding

relaxation, in the sense defined by [4]. Assuming the UGC, uselessness only holds for a limited class of predicates f (when $f^{-1}(1)$ supports a balanced pairwise independent distribution on $[q]^k$) [4]. Thus, proving an SDP lower bound for predicates which are not expected to be useless requires a new construction of instances, which cannot be generated uniformly at random. Our construction provides such a generalization, and may be useful in proving new SDP lower bounds. The properties of random $\mathcal{G}_{n,p}$ hypergraphs easily carry over to our instances, and we collect these properties in Section 3.

The above construction ensures that if the instance Φ_0 does not have an assignment satisfying more than an s fraction of the constraints, then $\mathsf{OPT}(\Phi) \leq s + \varepsilon$ with high probability. Also, it is well-known that providing a good LP solution to the relaxation given by t levels of the Sherali-Adams hierarchy is equivalent to providing distributions $\mathcal{D}_S$ on $[q]^S$ for all sets of variables S with $|S| \leq t$, such that the distributions are consistent restricted to subsets i.e., for all S with $|S| \leq t$ and all $T \subseteq S$, we have $\mathcal{D}_{S|T} = \mathcal{D}_T$. Thus, in our case, we need to produce such consistent local distributions such that the expected probability that a random constraint $C \in \Phi$ is satisfied by the local distribution on the set of variables involved in C (which we denote as S_C) is at least $c - \varepsilon$.

1.4.2 Local distributions from local structure

Most works on integrality gaps for CSPs utilize the local structure of random hypergraphs to produce such distributions. Since the girth of a sparse random hypergraph is $\Omega(\log n)$, any induced subgraph on $o(\log n)$ vertices is simply a forest. In case the induced (hyper)graph G_S on a set S is a *tree*, there is an easy distribution to consider: simply choose an arbitrary root and propagate down the tree by sampling each child conditioned on its parent. It is also easy to see that for $T \subseteq S$, if the induced (hyper)graph G_T is a *subtree* of G_S, then the distributions $\mathcal{D}_S$ and $\mathcal{D}_T$ produced as above are consistent.

The extension of this idea to forests requires some care. One can consider extending the distribution to forests by propagating independently on each tree in the forest. However, if for $T \subseteq S$ G_T is a forest while G_S is a tree, then a pair of vertices disconnected in G_T will have no correlation in $\mathcal{D}_T$ but may be correlated in $\mathcal{D}_S$. This was handled, for example, in [18] by adding noise to the propagation and using a large ball $B(S)$ around S to define $\mathcal{D}_S$. Then, if two vertices of T are disconnected in $B(T)$ but connected in $B(S)$, then they must be at a large distance from each other. Thus, because of the noise, the correlation between them (which is zero in $\mathcal{D}_T$) will be very small in $\mathcal{D}_S$. However, correcting approximate consistency to exact consistency incurs a cost which is exponential in the number of levels (i.e., the sizes of the sets), which is what limits the results in [18, 13] to $O(\log \log n)$ levels. This also makes the proof more involved since it requires a careful control of the errors in consistency.

1.4.3 Consistent partitioning schemes

We resolve the above consistency issue by first partitioning the given set S into a set of clusters, each of which have diameter $\Delta_H = o(\log n)$ in the underlying hypergraph H. Since each cluster has bounded diameter, it becomes a tree when we add all the missing paths between any two vertices in the cluster. We then propagate independently on each cluster (augmented with the missing paths). This preserves the correlation between any two vertices in the same cluster, even if the path between them was not originally present in G_S.

Of course, the above plan requires that the partition obtained for $T \subseteq S$, is consistent with the restriction to T of partition obtained for the set S. In fact, we construct distributions over

partitions $\{\mathcal{P}_S\}_{|S|\leq t}$, which satisfy the consistency property $\mathcal{P}_{S|T} = \mathcal{P}_T$. These distributions over partitions, which we call consistent partitioning schemes, are constructed in Section 4.

In addition to being consistent, we require that the partitioning scheme cuts only a small number of edges in expectation, since these contribute to a loss in the LP objective. We remark that such low-diameter decompositions (known as *separating* and *padded* decompositions) have been used extensively in the theory metric embeddings (see e.g., [22] and the references therein). The only additional requirement in our application is consistency.

We obtain the decompositions by proving the (easy) hypergraph extensions the results of Charikar, Makarychev and Makarychev [10], who exhibit a metric which is similar to the shortest path metric on graphs at small distances, and has the property that its restriction to any subset of size at most $n^{\varepsilon'}$ (for an appropriate $\varepsilon' < 1$) is ℓ_2 embeddable. This is proved in Section 3. A variant of this metric was used by Charikar, Makarychev and Makarychev [12] to prove lower bounds for MAX-CUT, for $n^{\varepsilon'}$ levels of the Sherali-Adams hierarchy. They used the embedding to construct a "local SDP solution" for any $n^{\varepsilon'}$ variables (with value $1 - \varepsilon'$) and produced the distributions required for Sherali-Adams by rounding the SDP solutions (which gives value $1 - O(\sqrt{\varepsilon'})$). However, rounding an SDP solution with a high value does not always produce a good integral solution for other CSPs.

Instead, we use these metrics in Section 4 to construct the consistent partitioning schemes as described above, by applying a result of Charikar et al. [9] giving separating decompositions for finite subsets of ℓ_2. We remark that it is the consistency requirement of the partitioning procedure that limits our results to $O\left(\frac{\log n}{\log\log n}\right)$ levels. The separation probability in the decomposition procedure grows with the dimension of the ℓ_2 embedding, while (to the best of our knowledge) dimension reduction procedures seem to break consistency.

2 Preliminaries

We use $[n]$ to denote the set $\{1, \ldots, n\}$. The only exception is $[q]$, where we overload this notation to denote the set $\{0, \ldots, q-1\}$, which corresponds to the the alphabet for the Constraint Satisfaction Problem under consideration. We use $\mathcal{D}_S$ and $\mathcal{P}_S$ to denote probability distributions over (assignments to or partitions of) a set S. For $T \subseteq S$, the notation $\mathcal{D}_{S|T}$ is used to denote the restriction (marginal) of the distribution $\mathcal{D}_S$ to the set T (and similarly for $\mathcal{P}_{S|T}$).

2.1 Constraint Satisfaction Problems

▶ **Definition 4.** Let $[q]$ denote the set $\{0, \ldots, q-1\}$. For a predicate $f : [q]^k \to \{0, 1\}$, an instance Φ of MAX k-CSP$_q$ (f) consists of a set of variables $\{x_1, \ldots, x_n\}$ and a set of constraints $C_1, \ldots, C_m$. Each constraint C_i is over a k-tuple of variables $(x_{i_1}, \ldots, x_{i_k})$ and is of the form

$$C_i \equiv f(x_{i_1} + b_{i_1}, \ldots, x_{i_k} + b_{i_k})$$

for some $b_{i_1}, \ldots, b_{i_k} \in [q]$, where the addition is modulo q. For an assignment $\sigma : \{x_1, \ldots, x_n\} \mapsto [q]$, let $\mathsf{sat}(\sigma)$ denote the fraction of constraints satisfied by σ. The maximum fraction of constraints that can be simultaneously satisfied is denoted by $\mathsf{OPT}(\Phi)$, i.e.

$$\mathsf{OPT}(\Phi) = \max_{\sigma:\{x_1,\ldots,x_n\}\mapsto[q]} \mathsf{sat}(\sigma).$$

For a constraint C of the above form, we use x_C to denote the tuple of variables $(x_{i_1}, \ldots, x_{i_k})$ and b_C to denote the tuple $(b_{i_1}, \ldots, b_{i_k})$. We then write the constraint as

$$
\begin{aligned}
\text{maximize} \quad & \mathop{\mathbb{E}}_{C\in\Phi}\left[\sum_{\alpha\in[q]^k} f(\alpha\cdot b_C)\cdot x_{(S_C,\alpha)}\right] & \\
\sum_{\substack{\alpha\in[q]^S\\ \alpha|_T=\beta}} x_{(S,\alpha)} &= x_{(T,\beta)} & \forall T\subseteq S\subseteq[n], |S|\le t,\ \forall\beta\in[q]^T\\
x_{(S,\alpha)} &\ge 0 & \forall S\subseteq[n], |S|\le t,\ \forall\alpha\in[q]^S\\
x_{(\emptyset,\emptyset)} &= 1 &
\end{aligned}
$$

Figure 1 Level-t Sherali-Adams LP for MAX k-CSP$_q$ (f).

$$
\begin{aligned}
\text{maximize} \quad & \mathop{\mathbb{E}}_{C\in\Phi}\left[\sum_{\alpha\in[q]^k} f(\alpha+b_C)\cdot x_{(S_C,\alpha)}\right] & \\
\sum_{j\in[q]} x_{(i,b)} &= 1 & \forall i\in[n]\\
\sum_{\substack{\alpha\in[q]^{S_C}\\ \alpha(i)=b}} x_{(S_C,\alpha)} &= x_{(i,b)} & \forall C\in\Phi, i\in S_C, b\in[q]\\
x_{(S_C,\alpha)} &\ge 0 & \forall C\in\Phi,\ \forall\alpha\in[q]^{S_C}
\end{aligned}
$$

Figure 2 Basic LP relaxation for MAX k-CSP$_q$ (f).

$f(x_C+b_C)$. We also denote by S_C the set of indices $\{i_1,\ldots,i_k\}$ of the variables participating in the constraint C.

2.2 The LP Relaxations for Constraint Satisfaction Problems

Below we present various LP relaxations for the MAX k-CSP$_q$ (f) problem that are relevant in this paper.

We start with the level-t Sherali-Adams relaxation. The intuition behind it is the following. Note that an integer solution to the problem can be given by an assignment $\sigma:[n]\to[q]$. Using this, we can define $\{0,1\}$-valued variables $x_{(S,\alpha)}$ for each $S\subseteq[n], 1\le|S|\le t$ and $\alpha\in[q]^S$, with the intended solution $x_{(S,\alpha)}=1$ if $\sigma(S)=\alpha$ and 0 otherwise. We also introduce a variable $x_{(\emptyset,\emptyset)}$, which equals to 1. We relax the integer program and allow variables to take real values in $[0,1]$. Now the variables $\{x_{(S,\alpha)}\}_{\alpha\in[q]^S}$ give a probability distribution $\mathcal{D}_S$ over assignments to S. We can enforce consistency between these *l*ocal distributions by requiring that for $T\subseteq S$, the distribution over assignments to S, when marginalized to T, is precisely the distribution over assignments to T i.e., $\mathcal{D}_{S|T}=\mathcal{D}_T$. The relaxation is shown in Figure 1.

The basic LP relaxation is a reduced form of the above relaxation where only those variables $x_{(S,\alpha)}$ are included for which $S=S_C$ is the set of CSP variables for some constraint C. The consistency constraints are included only for singleton subsets of the sets S_C. Note that the all the constraints for the basic LP are implied by the relaxation obtained by level k of the Sherali-Adams hierarchy.

For an LP/SDP relaxation of MAX k-CSP$_q$, and for a given instance Φ of the problem, we denote by FRAC(Φ) the LP/SDP (fractional) optimum. A relaxation is said to have a (c,s)-integrality gap if there exists a CSP instance Φ such that FRAC$(\Phi)\ge c$ and OPT$(\Phi)<s$.

2.3 Hypergraphs

An instance Φ of MAX k-CSP defines a natural associated hypergraph $H = (V, E)$ with V being the set of variables in Φ and E containing one k-hyperedge for every constraint $C \in \Phi$. We remind the reader of the familiar notions of degree, paths, and cycles for the case of (k-uniform) hypergraphs:

▶ **Definition 5.** Let $H = (V, E)$ be a hypergraph.

- For a vertex $v \in V$, the degree of the vertex v is defined to be the number of distinct hyperedges containing it.
- A simple path P is a finite alternate sequence of distinct vertices and distinct edges starting and ending at vertices, i.e., $P = v_1, e_1, v_2, \ldots, v_l, e_l, v_{l+1}$, where $v_i \in V\ \forall i \in [l+1]$ and $e_i \in E\ \forall i \in [l]$. Furthermore, e_i contains v_i, v_{i+1} for each i. Here l is called the length of the path P. All paths discussed in this paper will be simple paths.
- A sequence $\mathcal{C} = (v_1, e_1, v_2, \ldots, v_l, e_l, v_1)$ is called a cycle of length l if the initial segment $v_1, e_1, \ldots, v_l$ is a (simple) path, $e_{l+1} \neq e_i$ for all $i \in [l]$, and $v_1 \in e_l$. For a path P (or cycle $\mathcal{C}$), we use $\mathrm{V}(P)$ (or $\mathrm{V}(\mathcal{C})$) to denote the set of all the vertices that occur in the edges, i.e., the set $\{v\ :\ (\exists i \in [h])(v \in e_i)\}$, where $e_1, \ldots, e_h$ are the hyperedges included in P (or $\mathcal{C}$).
- For a given hypergraph H, the length of the smallest cycle in H is called the girth of H.

To observe the difference the notions of cycle in graphs and hypergraphs, it is instructive to consider the following example: let u, v be two distinct vertices in a k-uniform hypergraph for $k \geq 3$, and let e_1, e_2 be two distinct hyperedges both containing u and v. Then u, e_1, v, e_2, u is a cycle of length 2, which cannot occur in a graph.

We shall also need the following notion of the *closure* of a set $S \subseteq V$ in a given hypergraph H, defined by [12] for the case of graphs. A stronger notion of closure was also considered by [5].

▶ **Definition 6.** For a given hypergraph H and $R \in \mathbb{N}$, and a set $S \subseteq \mathrm{V}(H)$, we denote by $\mathrm{cl}_R(S)$ the R-closure of S obtained by adding all the vertices in all the paths of length at most R connecting two vertices of S, i.e.,

$$\mathrm{cl}_R(S) = S \cup \bigcup_{\substack{P : P \text{ is a path in H} \\ P \text{ connects } u,v \in S \\ |P| \leq R}} \mathrm{V}(P)\,.$$

For ease of notation, we use $\mathrm{cl}(S)$ to denote $\mathrm{cl}_1(S)$.

3 Properties of random hypergraphs

In this section we collect various properties of the hypergraphs corresponding to our integrality gap instances. The gap instances we generate contain several disjoint collections of variables. Each constraint in the instance has a specified "type", which specifies which of the collections each of the participating k variables much be sampled from. The constraint is generated by randomly sampling each of the k variables, from the collections specified by its type. This is captured by the generative model described below.

In the model below and in the construction of the gap instance, the parameter n_0 should be thought of as constant, while the parameters n and m should be though of a growing to infinity. We will choose $m = \gamma \cdot n$ for $\gamma = O_{k,q}(1)$.

▶ **Definition 7.** Let $n_0, k \in \mathbb{N}$ with $k \geq 2$. Let $m, n > 0$ and let Γ be a distribution on $[n_0]^k$. We define a distribution $\mathcal{H}_k(m, n, n_0, \Gamma)$ on k-uniform n_0-partite hypergraphs with m edges and $N = n_0 \cdot n$ vertices, divided in n_0 sets $X_1, \ldots, X_{n_0}$ of size n each. A random hypergraph $H \sim \mathcal{H}_k(m, n, n_0, \Gamma)$ is generated by sampling m random hyperedges independently as follows:

- Sample a random type $(i_1, \ldots, i_k) \in [n_0]^k$ from the distribution Γ.
- For all $j \in [k]$, sample v_{i_j} independently and uniformly in X_{i_j}.
- Add the edge $e_i = \{v_{i_1}, \ldots, v_{i_k}\}$ to H.

Note that as specified above, the model may generate a multi-hypergraph. However, the number of such repeated edges is likely to be small, and we will bound these, and in fact the number of cycles of size $o(\log n)$ in Lemma 26.

We will study the following metrics (similar to the ones defined in [10]):

▶ **Definition 8.** Given a hypergraph H with vertex set V, we define two metrics $d_\mu^H(\cdot,\cdot), \rho_\mu^H(\cdot,\cdot)$ on V as

$$d_\mu^H(u,v) \;\coloneqq\; 1 - (1-\mu)^{2 \cdot d_H(u,v)} \qquad \text{and} \qquad \rho_\mu^H(u,v) \;\coloneqq\; \sqrt{\frac{2 \cdot d_\mu^H(u,v) + \mu}{1+\mu}},$$

for $u \neq v$, where $d_H(\cdot,\cdot)$ denotes the shortest path distance in H.

We primarily need the fact the local ℓ_2-embeddability of the metric ρ_μ. The following theorem captures various properties of random hypergraphs required for our construction. The proof of the theorem heavily uses results proved in [3] and [12] and we defer the details to Appendix A.

▶ **Theorem 9.** *Let $H' \sim \mathcal{H}_k(m, n, n_0, \Gamma)$ with $m = \gamma \cdot n$ edges and let $\varepsilon > 0$. Then for large enough n, with high probability (at least $1 - \varepsilon$, over the choice of H'), there exists $\delta > 0$, constant $c = c(k, \gamma, n_0, \varepsilon)$, $\theta = \theta(k, \gamma, n_0, \varepsilon)$ and a subhypergraph $H \subset H'$ with $V(H) = V(H')$ satisfying the following:*

- *H has girth $\mathsf{g} \geq \delta \cdot \log n$.*
- *$|\mathrm{E}(H') \setminus \mathrm{E}(H)| \leq \varepsilon \cdot m$.*
- *For all $t \leq n^\theta$, for $\mu \geq c \cdot \frac{\log t + \log\log n}{\log n}$, for all $S \subseteq \mathrm{V}(H')$ with $|S| \leq t$, the metric ρ_μ^H restricted to S is isometrically embeddable into the unit sphere in ℓ_2,*

4 Decompositions of hypergraphs from local geometry

We will construct the Sherali-Adams solution by partitioning the given subset of vertices in to trees, and then creating a natural distribution over satisfying assignments on trees. We define below the kind of partitions we need.

▶ **Definition 10.** Let X be a finite set. For a set S, let $\mathcal{P}_S$ denote a distribution over partitions of S. For $T \subseteq S$, let $\mathcal{P}_{S|T}$ be the distribution over partitions of T obtained by restricting the partitions in $\mathcal{P}_S$ to the set T. We say that a collection of distributions $\{\mathcal{P}_S\}_{|S| \leq t}$ forms a consistent partitioning scheme of order t, if

$$\forall S \subseteq X, |S| \leq t \text{ and } \forall T \subseteq S \qquad \mathcal{P}_T = \mathcal{P}_{S|T}\,.$$

In addition to being consistent as described above, we also require the distributions to have small probability of cutting the hyperedges for the hypergraphs corresponding to our CSP instances. We define this property below.

▶ **Definition 11.** Let $H = (V, E)$ be a k-uniform hypergraph. Let $\{\mathcal{P}_S\}_{|S|\leq t}$ be a consistent partitioning scheme of order t for the vertex set V, with $t \geq k$. We say the scheme $\{\mathcal{P}_S\}_{|S|\leq t}$ is ε-sparse for H if

$$\forall e \in E \qquad \mathop{\mathbb{P}}_{P\sim\mathcal{P}_e}[P \neq \{e\}] \;\leq\; \varepsilon\,.$$

In this section, we will prove that the hypergraphs arising from random CSP instances admit sparse and consistent partitioning schemes. Recall that for a hypergraph H, we define (Definition 8) the metrics d_μ^H and ρ_μ^H as:

$$d_\mu^H(u,v) \;:=\; 1-(1-\mu)^{2\cdot d_H(u,v)} \qquad \text{and} \qquad \rho_\mu^H(u,v) \;:=\; \sqrt{\frac{2\cdot d_\mu^H(u,v)+\mu}{1+\mu}}\,,$$

▶ **Lemma 12.** *Let $H = (V, E)$ be k-uniform hypergraph and let d_μ be the metric as defined above. Let H be such that for all sets $S \subseteq V$ with $|S| \leq t$, the metric induced on ρ_μ on S is isometrically embeddable into ℓ_2. Then, there exists $\varepsilon \leq 10k \cdot \sqrt{\mu \cdot t}$ and $\Delta_H = O(1/\mu)$ such that H admits an ε-sparse consistent partitioning scheme of order t, with each partition consisting of clusters of diameter at most Δ_H in H.*

We use the following result of Charikar et al. [9] which shows that low-dimensional metrics have good *separating decompositions* with bounded diameter, i.e., decompositions which have a small probability of separating points at a small distance.

▶ **Theorem 13** ([9])**.** *Let W be a finite collection of points in $\mathbb{R}^d$ and let $\Delta > 0$ be given. Then there exists a distribution $\mathcal{P}$ over partitions of W such that*

- *$\forall P \in Supp(\mathcal{P})$, each cluster in P has ℓ_2 diameter at most Δ.*
- *For all $x, y \in W$*

$$\mathop{\mathbb{P}}_{P\sim\mathcal{P}}[P \text{ separates } x \text{ and } y] \;\leq\; 2\sqrt{d}\cdot\frac{\|x-y\|_2}{\Delta}\,.$$

We also need the observation that the partitions produced by the above theorem are consistent, assuming the set S considered above lie in a fixed bounded set (using a trivial modification of the procedure in [9]). For the sequel, we use $B(x, \delta)$ to denote the ℓ_2 ball around x of radius δ and $B_H(u, r)$ to denote a ball of radius r around a vertex $u \in V(H)$. Thus,

$$B(x,\delta) \;:=\; \{y \;\mid\; \|x-y\|_2 \leq \delta\} \qquad \text{and} \qquad B_H(u,r) \;:=\; \{v \in V \;\mid\; d_H(u,v) \leq r\}\,.$$

The balls $B(S, \delta)$ and $B_H(S, r)$ are defined similarly.

▶ **Claim 14.** *Let S and T be sets such that $T \subseteq S$. Let $W_S = \{w_u\}_{u\in S}$ and $W_T = \{w'_u\}_{u\in T}$ be ℓ_2-embeddings of S and T satisfying $\phi(W_T) \subseteq W_S \subseteq B(0, R_0) \subset \mathbb{R}^d$, for some unitary transformation ϕ and $R_0 > 0$. Let $\mathcal{P}_S$ and $\mathcal{P}_T$ be distributions over partitions of S and T respectively, induced by partitions on W_S and W_T as given by Theorem 13. Then*

$$\mathcal{P}_{S|T} \;=\; \mathcal{P}_T\,.$$

Proof. The claim follows simply by considering (a trivial modification of) the algorithm of [9]. For a given set W and a parameter Δ, they produce a partition using the following procedure:

- Let $W' = W$.
- Repeat until $W' = \emptyset$

- Pick a random point x in $B(W, \Delta/2)$ according to the Haar measure. Let $C_x = B(x, \Delta/2) \cap W'$.
- If $C_x \neq \emptyset$, set $W' = W' \setminus C_x$. Output C_x as a cluster in the partition.

[9] show that the above procedure produces a distribution over partitions satisfying the conditions in Theorem 13. We simply modify the procedure to sample a random point x in $B(0, R_0 + \Delta/2)$ instead of $B(S, \Delta/2)$. This does not affect the separation probability of any two points, since the only non-empty clusters are still produced by the points in $B(S, \Delta/2)$.

Let P be a partition of S produced by the above procedure when applied to the point set W_S, and let P' be a random partition produced when applied to the point set $\phi(W_T)$. It is easy to see from the above procedure that the distribution $\mathcal{P}_T$ is invariant under a unitary transformation of W_T. By coupling the random choice of a point in $B(0, R_0 + \Delta/2)$ chosen at each step in the procedures applied to W_S and $\phi(W_T) \subseteq W_S$, we get that $P(T) = P'$ i.e., the partition P restricted to T equals P'. Thus, we get $\mathcal{P}_{S|T} = \mathcal{P}_T$. ◀

We can use the above to prove Lemma 12.

Proof of Lemma 12. Given a set S, let W_S be an ℓ_2 embedding of the metric ρ_μ restricted to S. Since, $|S| \leq t$, we can assume $W_S \in \mathbb{R}^t$. We apply partitioning procedure of Charikar et al. from Theorem 13 with $\Delta = 1/2$. From the definition of the metric ρ^H_μ, we get that there exists a $\Delta_H = O(1/\mu)$ such that $\rho^H_{u,v} \leq 1/2 \implies d_H(u, v) \leq \Delta_H$. Moreover, for u, v contained in an edge e, we have that $\rho_\mu(u, v) \leq \sqrt{5\mu}$ and hence the probability that u and v are separated is at most $10\sqrt{\mu \cdot t}$. Thus, the probability that any vertex in e is separated from u is at most $10k \cdot \sqrt{\mu \cdot t}$.

Finally, for any $S \subseteq T$, if W_S and W_T denote the corresponding ℓ_2 embeddings, by the rigidity of ℓ_2 we have that for $\phi(W_T) \subseteq W_S$ for some unitary transformation ϕ. Thus, by Claim 14, we get that this is a consistent partitioning scheme of order t. ◀

5 The Sherali-Adams Integrality Gaps construction

5.1 Integrality Gaps from the Basic LP

Recall that the basic LP relaxation for MAX k-CSP$_q$ (f) as given in Figure 2. In this section, we will prove Theorem 1. We recall the statement below.

▶ **Theorem 1.** *Let $f : [q]^k \to \{0, 1\}$ be any predicate. Let Φ_0 be a (c, s) integrality gap instance for basic LP relaxation of MAX k-CSP (f). Then for every $\varepsilon > 0$, there exists $c_\varepsilon > 0$ such that for infinitely many $N \in \mathbb{N}$, there exist $(c - \varepsilon, s + \varepsilon)$ integrality gap instances of size N for the LP relaxation given by $c_\varepsilon \cdot \frac{\log N}{\log \log N}$ levels of the Sherali-Adams hierarchy.*

Let Φ_0 be a (c, s) integrality gap instance for the basic LP relaxation for MAX k-CSP$_q$ (f) with n_0 variables and m_0 constraints. We use it to construct a new integrality gap instance Φ. The construction is similar to the gap instances constructed by Khot et al. [18] discussed in the next section. However, we describe this construction first since it's simpler. The procedure for constructing the instance Φ is described in Figure 3.

5.1.1 Soundness

We first prove that no assignment satisfies more than $s + \varepsilon$ fraction of constraints for the above instance.

Given: A (c, s) gap instance Φ_0 on n_0 variables, for the basic LP.

Output: An instance Φ with $N = n \cdot n_0$ variables and m constraints.

The variables are divided into n_0 sets $X_1, \ldots, X_{n_0}$, one for each variable in Φ_0. We generate m constraints independently at random as follows:

1. Sample a random constraint $C_0 \sim \Phi_0$. Let $S_{C_0} = \{i_1, \ldots, i_k\} \subseteq [n_0]$ denote the set of variables in this constraint.
2. For each $j \in [k]$, sample a random variable $x_{i_j} \in X_{i_j}$.
3. Add the constraint $f((x_{i_1}, \ldots, x_{i_k}) + b_{C_0})$ to the instance Φ.

Figure 3 Construction of the gap instance Φ.

▶ **Lemma 15.** *For every $\varepsilon > 0$, there exists $\gamma = \gamma(\varepsilon, n_0, q)$ such that for an instance Φ generated by choosing at least $\gamma \cdot n$ constraints independently at random as above, we have with probability $1 - \exp(-\Omega(n))$, $\mathsf{OPT}(\Phi) < s + \varepsilon$.*

Proof. Fix an assignment $\sigma \in [q]^N$. We will first consider $\mathbb{E}[\mathsf{sat}_\Phi(\sigma)]$ for a randomly generated Φ as above.

$$\begin{aligned}\mathop{\mathbb{E}}_{\Phi}[\mathsf{sat}_\Phi(\sigma)] &= \mathop{\mathbb{E}}_{C_0 \in \Phi_0} \mathop{\mathbb{E}}_{x_{i_1} \in X_{i_1}} \cdots \mathop{\mathbb{E}}_{x_{i_k} \in X_{i_k}} [f(\sigma(x_{i_1}) + b_{i_1}, \ldots, \sigma(x_{i_k}) + b_{i_k})] \\ &= \mathop{\mathbb{E}}_{C_0 \in \Phi_0} \mathop{\mathbb{E}}_{Z_1, \ldots Z_{n_0}} [f(Z_{C_0} + b_{C_0})] \,,\end{aligned}$$

where for each $i \in [n_0]$, Z_i is an independent random variable with the distribution

$$\mathbb{P}[Z_i = b] := \mathop{\mathbb{E}}_{x \in X_i} \left[\mathbb{1}_{\{\sigma(x) = b\}}\right] ,$$

and Z_{C_0} denotes the collection of variables in the constraint C_0, i.e., $Z_{C_0} = \{Z_i\}_{i \in S_{C_0}}$. Thus, the random variables $Z_1, \ldots, Z_{n_0}$ define a random assignment to the variables in Φ_0, which gives, for any σ

$$\mathop{\mathbb{E}}_{\Phi}[\mathsf{sat}_\Phi(\sigma)] = \mathop{\mathbb{E}}_{C_0 \in \Phi_0} \mathop{\mathbb{E}}_{Z_1, \ldots Z_{n_0}} [f(Z_{C_0} + b_{C_0})] < s \,.$$

Consider a randomly added constraint C to the instance Φ. We have that

$$\mathbb{P}[C(\sigma) = 1] = \mathop{\mathbb{E}}_{\Phi}[\mathsf{sat}_\Phi(\sigma)] < s \,,$$

for any fixed σ over random choice of the constraint C. Thus, for an instance Φ with m independently and randomly generated constraints, we have

$$\begin{aligned}\mathop{\mathbb{P}}_{\Phi}[\mathsf{sat}_\Phi(\sigma) \geq s + \varepsilon] &\leq \mathop{\mathbb{P}}_{\Phi}\left[\mathsf{sat}_\Phi(\sigma) \geq \mathop{\mathbb{E}}_{\Phi}[\mathsf{sat}_\Phi(\sigma)] + \varepsilon\right] \\ &= \mathop{\mathbb{P}}_{\Phi}\left[\mathop{\mathbb{E}}_{C \in \Phi}\left[\mathbb{1}_{\{C(\sigma)=1\}}\right] \geq \mathop{\mathbb{E}}_{\Phi}[\mathsf{sat}_\Phi(\sigma)] + \varepsilon\right] \\ &\leq \exp\left(-\Omega(\varepsilon^2 \cdot m)\right) .\end{aligned}$$

Taking a union bound over all assignments, we get

$$\mathop{\mathbb{P}}_{\Phi}[\exists \sigma \;\; \mathsf{sat}_\Phi(\sigma) \geq s + \varepsilon] \leq q^{n \cdot n_0} \cdot \exp\left(-\varepsilon^2 \cdot m\right) ,$$

which is at most $\exp(-\Omega(n))$ for $m = O(((\log q)/\varepsilon^2) \cdot n \cdot n_0)$. ◀

5.1.2 Completeness

To prove the completeness, we first observe that the instance Φ as constructed above is also a gap instance for the basic LP. We will then "boost" this hardness to many levels of the Sherali-Adams hierarchy.

▶ **Lemma 16.** *For every $\varepsilon > 0$, there exists $\gamma = \gamma(\varepsilon)$ such that for an instance Φ generated by choosing at least $\gamma \cdot n$ constraints independently at random as above, with probability $1 - \exp(-\Omega(n))$ there exist distributions $\overline{\mathcal{D}}_{S_C}$ over $[q]^{S_C}$ for each $C \in \Phi$, and distributions $\overline{\mathcal{D}}_i$ over $[q]$ for each variable $x_i \in [n \cdot n_0]$, satisfying*

- *For all $C \in \Phi$ and all $i \in S_C$, $\overline{\mathcal{D}}_{S_C|\{i\}} = \mathcal{D}_i$.*
- *The distributions satisfy $\mathbb{E}_{C \in \Phi} \mathbb{E}_{\alpha \sim \overline{\mathcal{D}}_{S_C}} [f(\alpha + b_C)] \geq c - \frac{\varepsilon}{10}$.*

Proof. For each $C_0 \in \Phi_0$ and each $j \in [n_0]$, let $\mathcal{D}^{(0)}_{S_{C_0}}$ and $\mathcal{D}^{(0)}_j$ denote the basic LP solution satisfying

$$\mathcal{D}^{(0)}_{S_{C_0}|j} = \mathcal{D}^{(0)}_j \ \ \forall C_0 \in \Phi_0 \ \forall j \in S_{C_0} \qquad \text{and} \qquad \mathop{\mathbb{E}}_{C_0 \in \Phi_0} \mathop{\mathbb{E}}_{\alpha \sim \mathcal{D}^{(0)}_{S_{C_0}}} [f(\alpha + b_{C_0})] \geq c \,.$$

Each constraint $C \in \Phi$ is sampled according to some constraint $C_0 \in \Phi_0$, and we take $\overline{\mathcal{D}}_{S_C} := \mathcal{D}^{(0)}_{S_{C_0}}$ for the corresponding contraint $C_0 \in \Phi_0$. Also, each variable x_i for $i \in [n_0 \cdot n]$, belongs to one of the sets X_j for $j \in [n_0]$, and we take $\overline{\mathcal{D}}_i := \mathcal{D}^{(0)}_j$ for the corresponding $j \in [n_0]$.

The consistency of the distributions follows immediately from the construction of the instance Φ. Let $C \in \Phi$ be any constraint and let C_0 be the corresponding constraint in Φ_0. If $S_{C_0} = (j_1, \ldots, j_k)$, then $S_C = (i_1, \ldots, i_k)$ where each $i_r \in \{j_r\} \times [n]$ for all $r \in [k]$. Thus, for any $r \in [k]$,

$$\overline{\mathcal{D}}_{S_C|i_r} = \mathcal{D}^{(0)}_{S_{C_0}|j_r} = \mathcal{D}^{(0)}_{j_r} = \overline{\mathcal{D}}_{i_r} \,.$$

To bound the objective value, we again consider its expectation over a randomly generated instance Φ. Let C be a random constraint added to Φ. Then, if we define $\overline{\mathcal{D}}_{S_C}$ as above for this constraint, we have

$$\mathop{\mathbb{E}}_{C} \mathop{\mathbb{E}}_{\alpha \in \overline{\mathcal{D}}_{S_C}} [f(\alpha + b_C)] = \mathop{\mathbb{E}}_{C_0 \in \Phi_0} \mathop{\mathbb{E}}_{\alpha \sim \mathcal{D}^{(0)}} [f(\alpha + b_{C_0})] \geq c \,.$$

Thus, the expected contribution of each constraint is at least c. The probability that the average of m constraints deviates by at least $\varepsilon/10$ from the expectation, is at most $\exp\left(-\Omega(\varepsilon^2 \cdot m)\right)$. There exists $\gamma = O(1/\varepsilon^2)$ such that for $m \geq \gamma \cdot n$, the probability is at most $\exp(-\Omega(n))$. ◀

To construct local distributions for the Sherali-Adams hierarchy, we will consider (a slight modification) the hypergraph H corresponding to the instance Φ. We first show that distributions on hyperedges of this hypergraph can be consistently propagated in a tree, provided they agree on intersecting vertices.

For a set $U \subseteq \mathrm{V}(H)$ in a hypergraph H, recall that $\mathrm{cl}(U)$ includes all paths of lengths at most 1 between any two vertices in U. Thus, $E(\mathrm{cl}(U)) = \{e \in E \mid |e \cap U| \geq 2\}$. Note that Lemma 16 implies that hyperedges forming a tree in H satisfy the hypothesis of Lemma 17 below.

▶ **Lemma 17.** *Let $H = (V, E)$ be a k-uniform hypergraph. Let $U \subseteq V$ and let the set of hyperedges $E(\mathrm{cl}(U))$ form a tree. For each $e \in E(\mathrm{cl}(U))$, let $\overline{\mathcal{D}}_e$ be a distribution on $[q]^e$ such that for any $u \in U$ and $e_1, e_2 \in E(\mathrm{cl}(U))$ such that $e_1 \cap e_2 = \{u\}$, we have $\overline{\mathcal{D}}_{e_1|u} = \overline{\mathcal{D}}_{e_2|u} = \overline{\mathcal{D}}_u$. Then,*

- *there exists a distribution $\overline{\mathcal{D}}_U$ on $[q]^U$ such that $\overline{\mathcal{D}}_{U|e\cap U} = \overline{\mathcal{D}}_{e|e\cap U}$ for all $e \in E(U)$.*
- *If $U' \subseteq U$ is such that the hyperedges in $E(\mathrm{cl}(U'))$ form a subtree of $E(\mathrm{cl}(U))$, then $\overline{\mathcal{D}}_{U|U'} = \overline{\mathcal{D}}_{U'}$.*

Proof. We define the distribution by starting with an arbitrary hyperedge and traversing the tree in an arbitrary order. Let $e_1, \ldots, e_r$ be a traversal of the hyperedges in $E(\mathrm{cl}(U))$ such that for all i, $|(\cup_{j<i} e_j) \cap e_i| = 1$. Let $U_0 = \cup_{j<i} e_j$ be the set of vertices for which we have already sampled an assignment and let e_i be the next hyperedge in the traversal, with u being the unique vertex in $e_i \cap U_0$. We sample an assignment to the vertices in e, conditioned on the value for the vertex u. Formally, we extend the distribution $\overline{\mathcal{D}}_{U_0}$ to $U_0 \cup e$ by taking, for any $\alpha \in [q]^{U_0 \cup e}$

$$\overline{\mathcal{D}}_{U_0\cup e}(\alpha) \;=\; \overline{\mathcal{D}}_{U_0}(\alpha(U_0)) \cdot \frac{\overline{\mathcal{D}}_e(\alpha(e))}{\overline{\mathcal{D}}_{e|u}(\alpha(u))} \;=\; \overline{\mathcal{D}}_{U_0}(\alpha(U_0)) \cdot \frac{\overline{\mathcal{D}}_e(\alpha(e))}{\overline{\mathcal{D}}_u(\alpha(u))}\,.$$

The above process defines a distribution $\overline{\mathcal{D}}_{\mathrm{cl}(U)}$ on $\mathrm{cl}(U)$, with

$$\overline{\mathcal{D}}_{\mathrm{cl}(U)}(\alpha) \;=\; \frac{\prod_{e\in E(U)} \overline{\mathcal{D}}_e(\alpha(e))}{\prod_{u\in \mathrm{cl}(U)} \left(\overline{\mathcal{D}}_u(\alpha(u))\right)^{\deg(u)-1}}\,.$$

In the above expression, we use $\deg(u)$ to denote the degree of vertex u in tree formed by the hyperedges in $E(\mathrm{cl}(U))$ i.e., $\deg(u) = |\{e \in E(\mathrm{cl}(U)) \mid u \in e\}|$. We then define the distribution $\overline{\mathcal{D}}_U$ as the marginalized distribution $\overline{\mathcal{D}}_{\mathrm{cl}(U)|U}$ i.e.,

$$\overline{\mathcal{D}}_U(\alpha) \;=\; \sum_{\substack{\beta\in[q]^{\mathrm{cl}(U)} \\ \beta(U)=\alpha}} \overline{\mathcal{D}}_{\mathrm{cl}(U)}(\beta)\,.$$

Note that the distribution $\overline{\mathcal{D}}_{\mathrm{cl}(U)}$ and hence also the distribution $\overline{\mathcal{D}}_U$ are independent of the order in which we traverse the hyperedges in $E(\mathrm{cl}(U))$. Also, since the above process samples each hyperedge according to the distribution $\overline{\mathcal{D}}_e$, we have that for any $e \in E(U)$, $\overline{\mathcal{D}}_{\mathrm{cl}(U)|e} = \overline{\mathcal{D}}_e$. Thus, also for any $e \in E(U)$, $\overline{\mathcal{D}}_{U|e\cap U} = \overline{\mathcal{D}}_{e|e\cap U}$.

Let $U' \subseteq U$ be any set such that $E(\mathrm{cl}(U'))$ forms a subtree of $E(\mathrm{cl}(U))$. Then there exists a traversal $e_1, \ldots, e_r$, and $i \in [r]$ such that $e_j \in E(\mathrm{cl}(U'))\ \forall j \le i$ and $e_j \notin E(\mathrm{cl}(U'))\ \forall j > i$. However, the distribution defined by the partial traversal $e_1, \ldots, e_i$ is precisely $\overline{\mathcal{D}}_{\mathrm{cl}(U')}$. Thus, we get that $\overline{\mathcal{D}}_{\mathrm{cl}(U)|\,\mathrm{cl}(U')} = \overline{\mathcal{D}}_{\mathrm{cl}(U')}$ which implies $\overline{\mathcal{D}}_{U|U'} = \overline{\mathcal{D}}_{U'}$. ◀

We can now prove the completeness for our construction using consistent decompositions.

▶ **Lemma 18.** *Let $\varepsilon > 0$ and let Φ be a random instance of MAX k-CSP$_q$ (f) generated by choosing $\gamma \cdot n$ constraints independently at random as above. Then, there is a $t = \Omega_{\varepsilon,k,n_0}\left(\frac{\log n}{\log\log n}\right)$, such that with probability $1-\varepsilon$ over the choice of Φ, there exist distributions $\{\mathcal{D}_S\}_{|S|\le t}$ satisfying:*

- *For all $S \subseteq V$ with $|S| \le t$, $\mathcal{D}_S$ is a distribution on $[q]^S$.*
- *For all $T \subseteq S \subseteq V$ with $|S| \le t$, $\mathcal{D}_{S|T} = \mathcal{D}_T$.*
- *The distributions satisfy*

$$\mathop{\mathbb{E}}_{C\in\Phi}\ \mathop{\mathbb{E}}_{\alpha_C\sim\mathcal{D}_{S_C}} [f(\alpha_C + b_C)] \;\ge\; c - \varepsilon\,.$$

Proof. By Theorem 9, we know that there exists δ such that with probability $1 - \varepsilon/4$, after removing a set of constraints C_B of size at most $(\varepsilon/4) \cdot m$, we can assume that the remaining instance has girth at least $\mathsf{g} = \delta \cdot \log n$. Also, there exists $\theta, c > 0$ such that for all $t \leq n^\theta$, the metric ρ_μ^H restricted to any set S of size at most t embeds isometrically into the unit sphere in ℓ_2, for all $\mu \geq c \cdot \frac{\log t + \log\log n}{\log n}$.

We choose $\mu = 2c \cdot \frac{\log\log n}{\log n}$ and $t = \frac{\varepsilon^2}{400k^2} \cdot \frac{1}{\mu}$ so that

$$\mu \;\geq\; c \cdot \frac{\log t + \log\log n}{\log n} \quad \text{and} \quad \sqrt{\mu \cdot t} \;\leq\; \frac{\varepsilon}{20k}\,.$$

Thus, by Lemma 12, H admits an $(\varepsilon/2)$-sparse partitioning scheme of order t with each cluster in the partition having diameter at most $\Delta_H = O(1/\mu)$. Let $\{\mathcal{P}_S\}_{|S| \leq t}$ denote this partitioning scheme.

Given a set S, the distribution $\mathcal{D}_S$ is a convex combination of several distributions $\mathcal{D}_{S,P}$, corresponding to different partitions P sampled from $\mathcal{P}_S$. We describe the distribution $\mathcal{D}_S$ by giving the procedure to sample an $\alpha \in [q]^S$. Given the set S with $|S| \leq t$:

- Sample a partition $P = (U_1, \ldots, U_r)$ from the distribution $\mathcal{P}_S$.
- For each set U_i, consider the set $\mathcal{C}(U_i)$ obtained by including the vertices contained in all the hyperedges in the shortest path between all $u, v \in U_i$. Note that since U_i has diameter at most Δ_H in H, $\mathcal{C}(U_i)$ is connected and in fact $\mathcal{C}(U) = \mathrm{cl}_{\Delta_H}(U)$. Also, since the each vertex in an included path is within distance at most $\Delta_H/2$ of an end-point, and U_i has diameter at most Δ_H, we know that the diameter of $\mathcal{C}(U_i)$ is at most $2 \cdot \Delta_H$. Hence, $\mathcal{C}(U_i)$ is a tree. Finally, we must have $\mathrm{cl}(\mathcal{C}(U_i)) = \mathcal{C}(U_i)$ since any additional path of length 1 would create a cycle of length at most $2 \cdot \Delta_H + 1$.

  Thus, by Lemma 16 and Lemma 17 (with probability at least $1 - \varepsilon/4$) there exists a distribution $\overline{\mathcal{D}}_{\mathcal{C}(U_i)}$ for each U_i, satisfying $\overline{\mathcal{D}}_{\mathcal{C}(U_i)|e} = \overline{\mathcal{D}}_e$ for all $e \in E(\mathcal{C}(U_i))$. Here, $\overline{\mathcal{D}}_e$ are the distributions given by Lemma 16, which form a solution to the basic LP for Φ, with value at least $c - \varepsilon/4$. For each U_i, define the distribution

  $$\mathcal{D}'_{U_i} \;:=\; \overline{\mathcal{D}}_{\mathcal{C}(U_i)|U_i}\,.$$

- Sample $\alpha \in [q]^S$ according to the distribution

  $$\mathcal{D}_{S,P} \;:=\; \mathcal{D}'_{U_1} \times \cdots \times \mathcal{D}'_{U_r}\,.$$

Thus, we have

$$\mathcal{D}_S \;:=\; \mathop{\mathbb{E}}_{P=(U_1,\ldots,U_r)\sim\mathcal{P}_S} \left[\prod_{i=1}^{r} \mathcal{D}'_{U_i}\right]\,,$$

where the distributions $\mathcal{D}'_{U_i}$ are defined as above.

We first prove the distributions are consistent on intersections i.e., $\mathcal{D}_{S|T} = \mathcal{D}_T$ for any $T \subseteq S$. Note that by Lemma 12, the distributions $\mathcal{P}_S$ and $\mathcal{P}_T$ satisfy $\mathcal{P}_{S|T} = \mathcal{P}_T$. Each partition $(U_1, \ldots, U_r)$ also produces a partition T. For ease of notation, we assume that the first (say) r' clusters have non-empty intersection with S. Let $V_i = U_i \cap T$ for $1 \leq i \leq r'$

($V_i = \emptyset$ for $i > r'$). Then, we have

$$\begin{aligned}\mathcal{D}_{S|T} &= \mathop{\mathbb{E}}_{P=(U_1,\ldots,U_r)\sim\mathcal{P}_S}\left[\prod_{i=1}^{r}\mathcal{D}'_{U_i|V_i}\right] = \mathop{\mathbb{E}}_{P=(U_1,\ldots,U_r)\sim\mathcal{P}_S}\left[\prod_{i=1}^{r'}\overline{\mathcal{D}}_{\mathcal{C}(U_i)|V_i}\right] \\ &= \mathop{\mathbb{E}}_{P=(U_1,\ldots,U_r)\sim\mathcal{P}_S}\left[\prod_{i=1}^{r'}\overline{\mathcal{D}}_{\mathcal{C}(V_i)|V_i}\right] \\ &= \mathop{\mathbb{E}}_{P'=(V_1,\ldots,V_{r'})\sim\mathcal{P}_T}\left[\prod_{i=1}^{r'}\overline{\mathcal{D}}_{\mathcal{C}(V_i)|V_i}\right].\end{aligned}$$

The second to last equality above uses the fact that $\mathcal{C}(V_i)$ is a subtree of $\mathcal{C}(U_i)$ and thus $\overline{\mathcal{D}}_{\mathcal{C}(U_i)|\mathcal{C}(V_i)} = \overline{\mathcal{D}}_{\mathcal{C}(V_i)}$ by Lemma 17. The last equality uses the fact that $\mathcal{P}_{S|T} = \mathcal{P}_T$ by Lemma 12.

We now argue that the LP solution corresponding to the above distributions $\{\mathcal{D}_S\}_{|S|\le t}$ has value at least $c - \varepsilon$. Recall that the value of the LP solution is given by

$$\mathop{\mathbb{E}}_{C\in\Phi}\ \mathop{\mathbb{E}}_{\alpha\sim\mathcal{D}_{S_C}}[f(\alpha + b_C)]\,.$$

Consider any constraint C in Φ, with the corresponding set of variables S_C and the corresponding hyperedge e. When defining the distribution $\mathcal{D}_{S_C}$, we will partition S_C according to the distribution $\mathcal{P}_{S_C}$. By Lemma 12 and our choice of parameters

$$\mathop{\mathbb{P}}_{P\sim\mathcal{P}_{S_C}}[P \neq \{S_C\}] \le 10k\cdot\sqrt{\mu\cdot t} \le \frac{\varepsilon}{2}\,.$$

For a constraint set which is not in the deleted set C_B, if the hyperedge e corresponding to the constraint C is not split by a partition P sampled according to $\mathcal{P}_{S_C}$, then by Lemma 17 $\mathcal{D}_{S_C,P} = \overline{\mathcal{D}}_{S_C}$. Here, $\overline{\mathcal{D}}_{S_C}$ is the distribution given by Lemma 16. Since f is Boolean, we have that for $C \notin C_B$,

$$\mathop{\mathbb{E}}_{\alpha\sim\mathcal{D}_{S_C}}[f(\alpha + b_C)] \ge \mathop{\mathbb{E}}_{\alpha\sim\overline{\mathcal{D}}_{S_C}}[f(\alpha + b_C)] - \frac{\varepsilon}{2}\,.$$

Using Lemma 16 again, we get

$$\begin{aligned}\mathop{\mathbb{E}}_{C\sim\Phi}\ \mathop{\mathbb{E}}_{\alpha\sim\mathcal{D}_{S_C}}[f(\alpha + b_C)] &\ge \mathop{\mathbb{E}}_{C\sim\Phi}\left[\left(1 - \mathbb{1}_{\{C\in C_B\}}\right)\cdot\left(\mathop{\mathbb{E}}_{\alpha\sim\overline{\mathcal{D}}_{S_C}}[f(\alpha + b_C)] - \frac{\varepsilon}{2}\right)\right] \\ &\ge \mathop{\mathbb{E}}_{C\sim\Phi}\ \mathop{\mathbb{E}}_{\alpha\sim\overline{\mathcal{D}}_{S_C}}[f(\alpha + b_C)] - \frac{\varepsilon}{2} - \mathop{\mathbb{E}}_{C\sim\Phi}\left[\mathbb{1}_{\{C\in C_B\}}\right] \\ &\ge c - \frac{\varepsilon}{4} - \frac{\varepsilon}{2} - \frac{\varepsilon}{4} \\ &\ge c - \varepsilon\,,\end{aligned}$$

where the penultimate inequality uses the fact that the fraction of constraints in the initially deleted set C_B is at most $\varepsilon/4$ (for large enough n). ◀

5.2 Integrality Gaps for resistant predicates

Let $f : \{0,1\}^k \to \{0,1\}$ be a boolean predicate and let $\rho(f) = \frac{f^{-1}(1)}{2^k}$ be the fractions of satisfying assignments to f. Then f is approximation resistant if it is hard to distinguish the

MAX-CSP instances on f between which are at least $1 - o(1)$ satisfiable vs which are at most $\rho(f) + o(1)$ satisfiable.

In [18] the authors introduce the notion of vanishing measure (on a polytope defined by f) and use it to characterize a variant of approximation resistance, called strong approximation resistance, assuming the Unique Games conjecture. They also show gave a *weaker* notion of vanishing measures, which they used to characterize strong approximation resistance for LP hierarchies. In particular, they proved that when the condition in their characterization is satisfied, there exists a $(1 - o(1), \rho(f) + o(1))$ integrality gap for $O(\log \log n)$ levels of Sherali-Adams hierarchy for predicates f. Here, we show that using Theorem 1, their result can be simplified and strengthened [1] to $O\left(\frac{\log n}{\log \log n}\right)$ levels.

Let us first recall some useful notation defined by Khot et al. [18] before we define the notion of vanishing measure:

▶ **Definition 19.** For a predicate $f : \{0,1\}^k \to \{0,1\}$, let $\mathcal{C}(f)$ be the convex polytope of *first* moments (biases) of distributions supported on satisfying assignments of f i.e.,

$$\mathcal{C}(f) \;:=\; \left\{\zeta \in \mathbb{R}^k \;\mid\; \forall i \in [k],\; \zeta_i = \mathop{\mathbb{E}}_{\alpha \sim \nu}[(-1)^{\alpha_i}],\;\; \mathrm{Supp}(\nu) \subseteq f^{-1}(1)\right\}.$$

For a measure Λ on $\mathcal{C}(f)$, $S \subseteq [k]$, $b \in \{0,1\}^S$ and permutation $\pi : S \to S$, let $\Lambda_{S,\pi,b}$ denote the induced measure on $\mathbb{R}^S$ by considering vectors with coordinates $\left\{(-1)^{b_{\pi(i)}} \cdot \zeta_{\pi(i)}\right\}_{i \in S}$, where $\zeta \sim \Lambda$.

We recall below the definition of vanishing measure for LPs from [18] (see Definition 1.3) :

▶ **Definition 20.** A measure Λ on $\mathcal{C}(f)$ is called vanishing (for LPs) if for every $1 \le t \le k$, the following signed measure

$$\mathop{\mathbb{E}}_{|S|=t} \; \mathop{\mathbb{E}}_{\pi : S \to S} \; \mathop{\mathbb{E}}_{b \in \{0,1\}^t} \left[\left(\prod_{i=1}^{t} (-1)^{b_i}\right) \cdot \hat{f}(S) \cdot \Lambda_{S,\pi,b}\right]$$

is identically 0. We say f has a vanishing measure if there exists a vanishing measure Λ on $\mathcal{C}(f)$.

In particular, they prove the following theorem:

▶ **Theorem 21.** *Let $f : \{0,1\}^k \to \{0,1\}$ be a k-ary boolean predicate that has a vanishing measure. Then for every $\varepsilon > 0$, there is a constant $c_\varepsilon > 0$ such that for infinitely may $N \in \mathbb{N}$, there exists an instance Φ of MAX k-CSP(f) on N variables satisfying the following:*

- $\mathsf{OPT}(\Phi) \;\le\; \rho(f) + \varepsilon$.
- *The optimum for the LP relaxation given by $c_\varepsilon \cdot \log \log N$ levels of Sherali-Adams hierarchy has* $\mathsf{FRAC}(\Phi) \ge 1 - O(k \cdot \sqrt{\varepsilon})$.

Combining this with our Theorem 1 already gives us the following stronger result:

▶ **Corollary 22.** *Let $f : \{0,1\}^k \to \{0,1\}$ be a k-ary boolean predicate that has a vanishing measure. Then for every $\varepsilon > 0$, there is a constant $c_\varepsilon > 0$ such that for infinitely may $N \in \mathbb{N}$, there exists an instance Φ of MAX k-CSP(f) on N variables satisfying the following:*

- *All integral assignment of Φ satisfies at most $\rho(f) + \varepsilon$ fraction of constraints.*
- *The LP relaxation given by $c_\varepsilon \cdot \frac{\log N}{\log \log N}$ levels of Sherali-Adams hierarchy has* $\mathsf{FRAC}(\Phi) \ge 1 - O(k\sqrt{\varepsilon})$.

[1] The LP integrality gap result of Khot et al. is in fact slightly stronger than stated above. They show that LP value is at least $1 - o(1)$ while there is no integer solution achieving a value outside the range $[\rho(f) - o(1), \rho(f) + o(1)]$. It is easy to see that the same also holds for the instance constructed here.

Let $n_0 = \lceil \frac{1}{\varepsilon} \rceil$. Partition the interval $[0,1]$ into $n_0 + 1$ disjoint intervals $I_0, I_1, \ldots, I_{n_0}$ where $I_0 = \{0\}$ and $I_i = (i-1/n_0, i/n_0]$ for $1 \le i \le n_0$. For each interval I_i, let X_i be a collection of n variables (disjoint from all X_j for $j \neq i$).

Generate m constraints independently according to the following procedure:

- Sample $\zeta \sim \Lambda$.
- For each $j \in [k]$, let i_j be the index of the interval which contains $|\zeta(j)|$. Sample uniformly a variable y_j from the set X_j.
- If $\zeta(j) < 0$, then negate y_j. If $\zeta(j) = 0$, then negate y_j w.p. $\frac{1}{2}$.
- Introduce the constraint f on the sampled k tuple of literals.

Figure 4 Sherali-Adams integrality gap instance for vanishing measure.

However, note that to apply Theorem 1, one only needs a gap for the basic LP, which is much weaker requirement than the $O(\log \log N)$-level gap given by Theorem 21. We observe below that the gap for the basic LP follows very simply from the construction by Khot et al. [18]. One can then directly use this gap for applying Theorem 1 instead of going through Theorem 21.

Khot et al. [18] use the probabilistic construction given in Figure 4, for a given $\varepsilon > 0$. The construction actually requires Λ to be a vanishing measure over the polytope $\mathcal{C}_\delta(f) := (1-\delta) \cdot \mathcal{C}(f)$, for $\delta = \sqrt{\varepsilon}$. However, since $\mathcal{C}_\delta(f)$ is simply a scaling of $\mathcal{C}(f)$, a vanishing measure over $\mathcal{C}(f)$ also gives a vanishing measure over $\mathcal{C}_\delta(f)$. Note that each $\zeta_0 \in \mathcal{C}(f)$ corresponds to a distribution ν_0 supported in $f^{-1}(1)$. For each $\zeta \in \mathcal{C}_\delta$, let $\zeta_0 = \frac{1}{1-\delta} \cdot \zeta$ be the point in $\mathcal{C}(f)$ with distribution ν_0. Then the distribution $\nu = (1-\delta) \cdot \nu_0 + \delta \cdot U_k$ (where U_k denotes the uniform distribution on $\{0,1\}^k$) satisfies $\forall i \in [k]\, \mathbb{E}_{\alpha \sim \nu}[(-1)^{\alpha_i}] = \zeta_i$.

They show for a sufficiently large constant γ, an instance Φ with $m = \gamma \cdot n$ constraints satisfies with high probability, that for all assignments σ, $|\mathsf{sat}_\Phi(\sigma) - \rho(f)| \le \varepsilon$ (see Lemma 4.4 in [18]). The proof is similar to that of of Lemma 15.

Additionally, we need the following claim from [18] (see Claim 4.7 there), which allows one to "round" coordinates of the vectors $\zeta \in \mathcal{C}_\delta(f)$ to the end-points of the intervals $I_0, \ldots, I_{n_0}$. This ensures that any two variables in the same collection X_i have the same bias. The proof of the claim follows simply from a hybrid argument. We include it in the appendix for completeness.

▶ **Claim 23.** *Let $\zeta \in \mathcal{C}_\delta(f)$ and let ν be the corresponding distribution supported in $f^{-1}(1)$ such that for all $i \in [k]$, we have $\zeta_i = \mathbb{E}_{\alpha \sim \nu}[(-1)^{\alpha_i}]$. Let $t_1, \ldots, t_k \in [0,1]$ be such that for all $i \in [k]$, $|t_i - |\zeta_i|| \le \varepsilon$ for $\varepsilon < \delta/2$. Then there exists a distribution ν' on $\{0,1\}^k$ such that*

$$\|\nu - \nu'\|_1 \;=\; O(k \cdot (\varepsilon/\delta)) \qquad \text{and} \qquad \forall i \in [k], \;\; \mathop{\mathbb{E}}_{\alpha \sim \nu'}[(-1)^{\alpha_i}] \;=\; \mathrm{sign}(\zeta_i) \cdot t_i \,.$$

We can now use the above to give a simplified proof of Corollary 22.

Proof of Corollary 22. Here we exhibit a solution of the basic LP Figure 2 for the instance given in Figure 4. For each variable y_j coming from the set X_j for $j \in \{0, 1, \ldots, n_0\}$, we set the bias t_j of the variable to be the rightmost point of the interval I_j i.e., set $x_{(y_j,-1)} = \frac{1}{2} \cdot \left(1 - \frac{i}{n_0}\right)$ and $x_{(y_j,1)} = \frac{1}{2} \cdot \left(1 + \frac{i}{n_0}\right)$.

For each constraint C of the form $f(y_{i_1} + b_1, \ldots, y_{i_k} + b_k)$, let $\zeta(C) \in \mathcal{C}_\delta(f)$ be the point used to generate it, and let $\nu(C)$ denote the corresponding distribution on $\{0,1\}^k$. By Claim 23, there exists a distribution $\nu'(C)$ such that $\|\nu(C) - \nu'(C)\|_1 = O(k\varepsilon/\delta)$ and such

that the biases of the *literals* satisfy $\mathbb{E}_{\alpha\sim\nu'(C)}[(-1)^{\alpha_j}] = \text{sign}(\zeta_j)\cdot t_{i_j}$, where t_{i_j} denotes the bias for the interval to which y_{i_j} belongs. When $t_{i_j} \neq 0$, we negate a variable only when $\text{sign}(\zeta_j) < 0$. Thus, we have $\mathbb{E}_{\alpha\sim\nu'(C)}\left[(-1)^{\alpha_j+b_j}\right] = t_{i_j}$, which is consistent with the bias given by the singleton variables $x_{(y_{i_j},1)}$ and $x_{(y_{i_j},-1)}$. We thus define the local distribution on the set S_C as $\mathcal{D}_{S_C}(\alpha) = (\nu'(C))(\alpha + b_C)$.

For all $C \in \Phi$, since $\zeta(C) \in \mathcal{C}_\delta(f)$, we have that $\mathbb{E}_{\alpha\sim\nu(C)}[f(\alpha)] \geq 1-\delta$. Also, since $\|\nu(C) - \nu'(C)\|_1 = O(k\varepsilon/\delta)$, we get that $\mathbb{E}_{\alpha\sim\nu'(C)}[f(\alpha)] \geq 1-\delta-O(k\varepsilon/\delta)$. Thus, we have for all $C \in \Phi$, $\mathbb{E}_{\alpha\sim\mathcal{D}_{S_C}}[f(\alpha+b_C)] \geq 1-\delta-O(k\varepsilon/\delta)$. Taking $\delta = \sqrt{\varepsilon}$ proves the claim. ◀

5.3 Lower bounds for LP extended formulations

A connection between LP integrality gaps for the Sheral-Adams hierarchy, and lower bounds on the size of LP extended formulations, was first established by Chan et al. [8] and later improved by Kothari et al. [20]. In [20], the authors proved the following:

▶ **Theorem 24** ([20], Theorem1.2)**.** *There exist constants $0 < h < H$ such that the following holds. Consider a function $f : \mathbb{N} \to \mathbb{N}$. Suppose that the $f(n)$-level Sherali-Adams relaxation for a CSP cannot achieve a (c,s)-approximation on instances on n variables. Then, no LP extended formulation (of the original LP) of size at most $n^{h\cdot f(n)}$ can achieve a (c,s)-approximation for the CSP on n^H variables.*

Combining Theorem 1 with Theorem 24 yields (with $f(N) = c_\varepsilon \cdot \frac{\log N}{\log\log N}$):

▶ **Corollary 2.** *Let $f : [q] \to \{0,1\}$ be any predicate. Let Φ_0 be a (c,s) integrality gap instance for basic LP relaxation of MAX k-CSP (f). Then for every $\varepsilon > 0$, there exists $c'_\varepsilon > 0$ such that for infinitely $N \in \mathbb{N}$, there exist $(c-\varepsilon, s+\varepsilon)$ integrality gap instances of size N, for every linear extended formulation of size at most $N^{c'_\varepsilon \cdot \frac{\log N}{\log\log N}}$.*

Acknowledgements. We thank Chandra Chekuri, Subhash Khot and Yury Makarychev for helpful discussions, and Rishi Saket for pointers to references.

References

1 Michael Alekhnovich, Sanjeev Arora, and Iannis Tourlakis. Towards strong nonapproximability results in the Lovasz-Schrijver hierarchy. In *Proceedings of the 37th ACM Symposium on Theory of Computing*, pages 294–303, 2005.

2 Sanjeev Arora, Béla Bollobás, and László Lovász. Proving integrality gaps without knowing the linear program. In *Proceedings of the 43rd IEEE Symposium on Foundations of Computer Science*, pages 313–322. IEEE, 2002. doi:10.1109/SFCS.2002.1181954.

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 Per Austrin and Johan Håstad. On the usefulness of predicates. *ACM Trans. Comput. Theory*, 5(1):1:1–1:24, May 2013. doi:10.1145/2462896.2462897.

5 Boaz Barak, Siu On Chan, and Pravesh K. Kothari. Sum of squares lower bounds from pairwise independence. In *Proceedings of the 47th ACM Symposium on Theory of Computing*, pages 97–106, New York, NY, USA, 2015. ACM. doi:10.1145/2746539.2746625.

6 Eli Ben-Sasson. *Expansion in Proof Complexity.* PhD thesis, Hebrew University, 2001.

7 Siavosh Benabbas, Konstantinos Georgiou, Avner Magen, and Madhur Tulsiani. SDP gaps from pairwise independence. *Theory of Computing*, 8(12):269–289, 2012. doi:10.4086/toc.2012.v008a012.

8 Siu On Chan, James R. Lee, Prasad Raghavendra, and David Steurer. Approximate constraint satisfaction requires large LP relaxations. In *Proceedings of the 54th IEEE Symposium on Foundations of Computer Science*, pages 350–359, Washington, DC, USA, 2013. IEEE Computer Society. doi:10.1109/FOCS.2013.45.

9 Moses Charikar, Chandra Chekuri, Ashish Goel, Sudipto Guha, and Serge Plotkin. Approximating a finite metric by a small number of tree metrics. In *Proceedings of the 39th IEEE Symposium on Foundations of Computer Science*, pages 379–388, 1998. doi:10.1109/SFCS.1998.743488.

10 Moses Charikar, Konstantin Makarychev, and Yury Makarychev. Local global tradeoffs in metric embeddings. In *Proceedings of the 48th IEEE Symposium on Foundations of Computer Science*, pages 713–723, Washington, DC, USA, 2007. IEEE Computer Society. doi:10.1109/FOCS.2007.37.

11 Moses Charikar, Konstantin Makarychev, and Yury Makarychev. Near-optimal algorithms for maximum constraint satisfaction problems. In *Proceedings of the 18th ACM-SIAM Symposium on Discrete Algorithms*, pages 62–68, 2007. doi:10.1145/1283383.1283391.

12 Moses Charikar, Konstantin Makarychev, and Yury Makarychev. Integrality gaps for Sherali-Adams relaxations. In *Proceedings of the 41st ACM Symposium on Theory of Computing*, pages 283–292, New York, NY, USA, 2009. ACM. doi:10.1145/1536414.1536455.

13 Wenceslas Fernandez de la Vega and Claire Kenyon-Mathieu. Linear programming relaxations of maxcut. In *Proceedings of the 18th ACM-SIAM Symposium on Discrete Algorithms*, pages 53–61, Philadelphia, PA, USA, 2007. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=1283383.1283390.

14 Sreyash Kenkre, Vinayaka Pandit, Manish Purohit, and Rishi Saket. On the approximability of digraph ordering. In Nikhil Bansal and Irene Finocchi, editors, *Algorithms – ESA 2015: 23rd Annual European Symposium, Patras, Greece, September 14-16, 2015, Proceedings*, pages 792–803, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg. doi:10.1007/978-3-662-48350-3_66.

15 Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the 34th ACM Symposium on Theory of Computing*, pages 767–775, New York, NY, USA, 2002. ACM. doi:10.1145/509907.510017.

16 Subhash Khot and Rishi Saket. SDP Integrality Gaps with Local ℓ_1-Embeddability. In *Proceedings of the 50th IEEE Symposium on Foundations of Computer Science*, pages 565–574, Washington, DC, USA, 2009. IEEE Computer Society. doi:10.1109/FOCS.2009.37.

17 Subhash Khot and Rishi Saket. Approximating CSPs using LP relaxation. In *Proceedings of the 42nd International Colloquium on Automata, Languages and Programming*, volume 9134, pages 822–833. Springer Verlag, 2015. doi:10.1007/978-3-662-47672-7_67.

18 Subhash Khot, Madhur Tulsiani, and Pratik Worah. A characterization of strong approximation resistance. In *Proceedings of the 46th ACM Symposium on Theory of Computing*, pages 634–643, New York, NY, USA, 2014. ACM. doi:10.1145/2591796.2591817.

19 Vladimir Kolmogorov, Johan Thapper, and Stanislav Živný. The power of linear programming for general-valued CSPs. *SIAM Journal on Computing*, 44(1):1–36, 2015. doi:10.1137/130945648.

20 Pravesh 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 ACM Symposium on Theory of Computing*, 2017. (To appear).

21 Pravesh Kothari, Ryuhei Mori, Ryan O'Donnell, and David Witmer. Sum of squares lower bounds for refuting any CSP. In *Proceedings of the 49th ACM Symposium on Theory of Computing*, 2017. (To appear).

22 Robert Krauthgamer, James R Lee, Manor Mendel, and Assaf Naor. Measured descent: A new embedding method for finite metrics. *Geometric & Functional Analysis GAFA*, 15(4):839–858, 2005. doi:10.1007/s00039-005-0527-6.

23 Euiwoong Lee. Hardness of graph pricing through generalized max-dicut. In *Proceedings of the 47th ACM Symposium on Theory of Computing*, pages 391–399, New York, NY, USA, 2015. ACM. doi:10.1145/2746539.2746549.

24 L. Lovász and A. Schrijver. Cones of matrices and set-functions and 0-1 optimization. *SIAM J. on Optimization*, 1(12):166–190, 1991. doi:10.1137/0801013.

25 Prasad Raghavendra. Optimal algorithms and inapproximability results for every CSP? In *Proceedings of the 40th ACM Symposium on Theory of Computing*, pages 245–254, New York, NY, USA, 2008. ACM. doi:10.1145/1374376.1374414.

26 Prasad Raghavendra and David Steurer. Integrality gaps for strong SDP relaxations of unique games. In *Proceedings of the 50th IEEE Symposium on Foundations of Computer Science*, 2009. doi:10.1109/FOCS.2009.73.

27 Grant Schoenebeck. Linear level Lasserre lower bounds for certain k-CSPs. In *Proceedings of the 49th IEEE Symposium on Foundations of Computer Science*, pages 593–602, Washington, DC, USA, 2008. IEEE Computer Society. doi:10.1109/FOCS.2008.74.

28 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.

29 Johan Håstad. On the Efficient Approximability of Constraint Satisfaction Problems. In *Surveys in Combinatorics*, volume 346, pages 201–222. Cambridge University Press, 2007. doi:10.1017/CBO9780511666209.008.

30 Johan Thapper and Stanislav Živný. The complexity of finite-valued CSPs. In *Proceedings of the 45th ACM Symposium on Theory of Computing*, pages 695–704, New York, NY, USA, 2013. ACM. doi:10.1145/2488608.2488697.

31 Johan Thapper and Stanislav Živný. The power of Sherali-Adams relaxations for general-valued CSPs. *arXiv preprint arXiv:1606.02577*, 2016.

32 Madhur Tulsiani. CSP gaps and reductions in the Lasserre hierarchy. In *Proceedings of the 41st ACM Symposium on Theory of Computing*, pages 303–312, New York, NY, USA, 2009. ACM. doi:10.1145/1536414.1536457.

A Local ℓ_2-embeddability of the metric ρ_μ^H

The goal of this section is to prove the following result about the local ℓ_2-embeddability of the metric ρ_μ^H.

▶ **Theorem 9.** *Let $H' \sim \mathcal{H}_k(m, n, n_0, \Gamma)$ with $m = \gamma \cdot n$ edges and let $\varepsilon > 0$. Then for large enough n, with high probability (at least $1 - \varepsilon$, over the choice of H'), there exists $\delta > 0$, constant $c = c(k, \gamma, n_0, \varepsilon)$, $\theta = \theta(k, \gamma, n_0, \varepsilon)$ and a subhypergraph $H \subset H'$ with $V(H) = V(H')$ satisfying the following:*

- *H has girth $\mathsf{g} \geq \delta \cdot \log n$.*
- *$|\mathrm{E}(H') \setminus \mathrm{E}(H)| \leq \varepsilon \cdot m$.*
- *For all $t \leq n^\theta$, for $\mu \geq c \cdot \frac{\log t + \log\log n}{\log n}$, for all $S \subseteq \mathrm{V}(H')$ with $|S| \leq t$, the metric ρ_μ^H restricted to S is isometrically embeddable into the unit sphere in ℓ_2,*

To prove the above theorem, we will use the local structure of random hypergraphs. We first prove that with high probability for random hypergraphs (sampled from $\mathcal{H}_k(m, n, n_0, \Gamma)$) a few hyperedges can be removed to obtain a hypergraph whose girth is $\Omega(\log n)$ and the degree is bounded. The following lemma shows a possible trade-off between the degree of the hypergraph vs the number of hyperedges required to be removed.

▶ **Lemma 25.** *Let $H' \sim \mathcal{H}_k(m, n, n_0, \Gamma)$ be a random hypergraph with $m = \gamma \cdot n$ hyperedges. Then for any $\varepsilon > 0$, with probability $1-\varepsilon$, there exists a sub-hypergraph H with $V(H) = V(H')$ such that $\forall u \in V(H)$, $\deg_H(u) \leq 100 \cdot \log\left(\frac{n_0}{\varepsilon}\right) \cdot k \cdot \gamma$ and $|\mathrm{E}(H') \setminus \mathrm{E}(H)| \leq \varepsilon \cdot m$.*

Proof. By linearity of expectation, the expected degree of any vertex v in H' is at most $k \cdot \gamma$. Let $D = 100 \cdot \log\left(\frac{n_0}{\varepsilon}\right) \cdot k \cdot \gamma$, and let S be the set of all vertices u with $\deg_{H'}(u) > D$. Let E_S be the set of all hyperedges with at least one vertex in S. We shall take $E(H) = E(H') \setminus E_S$. Note that for any $u \in V(H')$, $\mathbb{P}[u \in S] = \mathbb{P}[\deg_{H'}(u) \geq D] \leq \exp(-D/4)$ by a Chernoff-Hoeffding bound. We use this to bound the expected number of edges deleted.

$$\begin{aligned}
\mathbb{E}[E_S] \;\leq\; \sum_{u \in V(H')} \mathbb{E}\left[\deg(u) \cdot \mathbb{1}_{\{u \in S\}}\right] \;&=\; \sum_{u \in V(H')} \mathbb{E}[\deg(u) \;\mid\; u \in S] \cdot \mathbb{P}[u \in S] \\
&\leq \sum_{u \in V(H')} \mathbb{E}[\deg(u) \;\mid\; u \in S] \cdot \exp(-D/4) \\
&\leq \sum_{u \in V(H')} (D + k\gamma) \cdot \exp(-D/4) \\
&\leq\; (n \cdot n_0) \cdot 2D \cdot \exp(-D/4)\ .
\end{aligned}$$

The penultimate inequality uses the independence of the hyperedges in the generation process, which gives $\mathbb{E}[\deg_{H'}(u) \mid \deg_{H'}(u) \geq D] \;\leq\; D + \mathbb{E}[\deg_{H'}(u)]$. From our choice of the parameter D, we get that $\mathbb{E}[E_S] \leq \varepsilon^2 \cdot \gamma \cdot n = \varepsilon^2 \cdot m$. Thus, the number of edges deleted is at most $\varepsilon \cdot m$ with probability at least $1 - \varepsilon$. ◀

The following lemma shows that the expected number of small cycles in random hypergraph is small.

▶ **Lemma 26.** *Let $H \sim \mathcal{H}_k(m, n, n_0, \Gamma)$ be a random hypergraph and for $l \geq 2$, let $Z_l(H)$ denote the number of cycles of length at most l in H. For m, n and k such that $k^2 \cdot (m/n) > 1$, we have*

$$\mathop{\mathbb{E}}_{H \sim \mathcal{H}_k(m,n,n_0,\Gamma)}[Z_l(H)] \leq \left(k^2 \cdot \frac{m}{n}\right)^{2l}\ .$$

Proof. Let the vertices of H correspond to the set $[n_0] \times [n]$. Suppose we contract the set of $[n_0] \times \{j\}$ vertices into a single vertex $j \in [n]$ to get a random multi-hypergraph H' on vertex set $[n]$. An equivalent way to view the sampling to H' is: for each $i \in [m]$, the i-th hyperedge e_i of H' is sampled by independently sampling k vertices (with replacement) uniformly at random from $[n]$. Note that the sampling of H' is independent of Γ in the definition of $\mathcal{H}_k(m, n, n_0, \Gamma)$. Clearly, a cycle of length at most l in H produces a cycle of length at most l in H'. Hence, it suffices to bound the expected number of cycles in H'

Given any pair (u', v') of vertices of H', for $u' \neq v'$, the probability of the pair (u', v') belonging together in some hyperedge of H' is at most $\frac{mk^2}{n^2}$. Consider a given h-tuple of vertices $\mathbf{u} = (u_{i_1}, \cdots u_{i_h})$. Note that we require that hyperedges participating in a cycle be distinct. So, the probability that $\mathbf{u}$ is part of a cycle in H', i.e., there exists distinct hyperedges $e_j \in H'$ for $j \in [h]$ such that $u_{i_j}, u_{i_{j+1}} \in e_j$ for $j \in [h-1]$, and $u_{i_1}, u_{i_h} \in e_h$ is at most $\left(\frac{mk^2}{n^2}\right)^h$. As a result, expected number of cycles of length h in H' is bounded above by:

$$\binom{n}{h}\left(\frac{mk^2}{n^2}\right)^h \leq n^h \left(\frac{mk^2}{n^2}\right)^h = \left(k^2 \cdot \frac{m}{n}\right)^h$$

From the geometric form of the bound, it follows that expected number of cycles of length at most l in H' is at most $\frac{\left(k^2 \cdot \frac{m}{n}\right)^{l+1}}{\left(k^2 \cdot \frac{m}{n}\right) - 1} < \left(k^2 \cdot \frac{m}{n}\right)^{2l}$. ◀

Using the above lemma, it is easy to show that one can remove all small cycles in a random hypergraph by deleting only a small number of hyperedges.

▶ **Corollary 27.** *Let $H \sim \mathcal{H}_k(m, n, n_0, \Gamma)$ be a random hypergraph with $m = \gamma \cdot n$ for $\gamma > 1$ and $k \geq 2$. Then, there exists $\delta = \delta(\gamma) > 0$ such that with probability $1 - n^{-1/6}$, all cycles of length at most $\delta \cdot \log n$ in H can be removed by deleting at most $n^{2/3}$ hyperedges.*

Proof. As above, let Z_l denote the number of cycles of length at most l. With the choice of m, n, and k, we have $k^2 \cdot \frac{m}{n} \geq 2$. By Lemma 26, $\mathbb{E}[Z_l] \leq \left(k^2 \cdot \frac{m}{n}\right)^{2l}$. Since $m = \gamma \cdot n$, there exists a $\mathsf{g} = \delta \cdot \log n$ such that $\mathbb{E}[Z_l] \leq \sqrt{n}$. By Markov's inequality, $\mathbb{P}\left[Z_l \geq n^{2/3}\right] \leq n^{-1/6}$. Thus, with probability $1 - n^{-1/6}$, one can remove all cycles of length at most $\delta \cdot \log n$ by deleting at most $n^{2/3}$ edges. ◀

One can also extend the analysis in [3] to show that the hypergraphs are locally sparse, i.e., the number of hyperedges contained in a small set of vertices is small. For a hypergraph H and a set $S \subseteq \mathrm{V}(H)$, we use $E(S)$ to denote the edges contained in the set S.

▶ **Definition 28.** We say that $S \subseteq V(H)$ is η-sparse if $|E(S)| \leq \frac{|S|}{k-1-\eta}$. We call a k-uniform hypergraph H on N vertices to be (τ, η)-sparse if all subsets $S \subset \mathrm{V}(H), |S| \leq \tau \cdot |\mathrm{V}(H)|$, S is η-sparse. We call H to be η-sparse if it is $(1, \eta)$-sparse, i.e., all subsets of vertices of H are sparse.

We note here that while this notion of sparsity is a generalization of that considered in [3], it is also identical to the notions of expansion considered in works in proof complexity (see e.g., [6]) and later in works on integrality gaps [1, 7, 5]. We prove that random hypergraphs generated with our model are locally sparse:

▶ **Lemma 29.** *Let $\eta < 1/4$ and $m = \gamma \cdot n$ for $\gamma > 1$. Then for $\tau \leq \frac{1}{n_0} \cdot \left(\frac{1}{e \cdot k^{3k} \cdot \gamma}\right)^{1/\eta}$ the following holds:*

$$\mathop{\mathbb{P}}_{H \sim \mathcal{H}_k(m,n,n_0,\Gamma)} [H \text{ is not } (\tau, \eta)\text{-sparse}] \leq 3 \cdot \left(\frac{k^{3k} \cdot \gamma}{n^{\eta/4}}\right)^{1/k}.$$

We note that we will require the sparsity η to be $O_{k,\gamma}(1/\log n)$. This gives sparsity only for sublinear size sets, as compared to sets of size $\Omega(n)$ in previous works where η is a constant. For the proof of the lemma, we follow an approach similar to that of Lemma 26: we collapse the vertices of H of the form $[n_0] \times \{j\}$ to vertex $j \in [n]$ to construct H', and thus reducing the problem to random multi-hypergraph form a random multipartite hypergraph. The rest proof of the lemma is along the lines of several known proofs [1, 7].

Proof. As in the proof of Lemma 26, given a random hypergraph H, we construct a hypergraph H' (by contracting all the vertices in $[n_0] \times \{j\}$ to $j \in [n]$).

Consider a subset of vertices $S \subseteq \mathrm{V}(H)$ and let $S' \subseteq \mathrm{V}(H')$ be the corresponding contracted set in H'. Since each edge in H corresponds to an edge in H' (counting multiplicities), we have

$$|E(S)| \geq \frac{|S|}{k-1-\eta} \quad \Rightarrow \quad |E(S')| \geq \frac{|S|}{k-1-\eta} \geq \frac{|S'|}{k-1-\eta}.$$

Thus, it suffices to show that H' is (τ', η)-sparse for $\tau' = \tau \cdot n_0$, since $|S'| \leq \tau \cdot N = (\tau \cdot n_0) \cdot n$. Given any multiset in $[n]^k$, the probability that it corresponds to an edge in H' is at most

$(k!) \cdot (m/n^k)$. Thus, the probability that there exists a set T of size at most $\tau' \cdot n$, containing at least $|T|/(k-1-\eta)$ edges (counting multiplicities) is at most

$$\sum_{h=1}^{\tau' \cdot n} \binom{n}{h} \cdot \binom{h^k}{r} \cdot \left(\frac{k! \cdot m}{n^k}\right)^r ,$$

where $r = \frac{h}{k-1-\eta}$. Note that we also need to consider $h = 1$ as edges in H' correspond to multisets of size k, and so may not have all distinct vertices. Simplifying the above using $\binom{a}{b} \leq \left(\frac{a \cdot e}{b}\right)^b$ and $k! \leq k^k$ gives

$$\begin{aligned}
\sum_{h=1}^{\tau' \cdot n} \binom{n}{h} \cdot \binom{h^k}{r} \cdot \left(\frac{k! \cdot m}{n^k}\right)^r &\leq \sum_{h=1}^{\tau' \cdot n} \left(\frac{n \cdot e}{h}\right)^h \cdot \left(\frac{h^k \cdot e}{r}\right)^r \cdot \left(\frac{k^k \cdot m}{n^k}\right)^r \\
&= \sum_{h=1}^{\tau' \cdot n} \left(e^{k-\eta} \cdot (k-1-\eta) \cdot k^k \cdot \gamma \cdot \left(\frac{h}{n}\right)^\eta\right)^{h/(k-1-\eta)} \\
&\leq \sum_{h=1}^{\tau' \cdot n} \left(k^{3k} \cdot \gamma \cdot \left(\frac{h}{n}\right)^\eta\right)^{h/(k-1-\eta)}
\end{aligned}$$

Let $\theta = \eta/(2k)$. We divide the above summation in two parts and first consider

$$\begin{aligned}
\sum_{h=n^\theta}^{\tau' \cdot n} \left(k^{3k} \cdot \gamma \cdot \left(\frac{h}{n}\right)^\eta\right)^{h/(k-1-\eta)} &\leq \sum_{h=n^\theta}^{\tau' \cdot n} \left(k^{3k} \cdot \gamma \cdot (\tau')^\eta\right)^{n^\theta/(k-1-\eta)} \\
&\leq 2 \cdot \exp\left(-\frac{n^\theta}{k}\right) \\
&\leq 2 \cdot \frac{k}{n^\theta} ,
\end{aligned}$$

for $\tau' \leq \left(e \cdot k^{3k} \cdot \gamma\right)^{-1/\eta}$. Considering the first half of the summation, we get

$$\begin{aligned}
\sum_{h=1}^{n^\theta} \left(k^{3k} \cdot \gamma \cdot \left(\frac{h}{n}\right)^\eta\right)^{h/(k-1-\eta)} &\leq n^\theta \cdot \left(\frac{k^{3k} \cdot \gamma}{n^{(1-\theta)\cdot\eta}}\right)^{1/k} \\
&\leq \left(\frac{k^{3k} \cdot \gamma}{n^{\eta/4}}\right)^{1/k} = k^3 \cdot \gamma^{1/k} \cdot n^{-\theta/2} .
\end{aligned}$$

Combining the two bounds gives that the probability is at most $3k^3 \cdot \gamma^{1/k} \cdot n^{-\theta/2}$, which equals the desired bound. ◀

Charikar et al. [10] prove an analogue of Theorem 9 for metrics defined on locally-sparse graphs. In fact, they use a consequence of sparsity, which they call ℓ-path decomposability. To this end, we define the *incidence graph*[2] associated with a hypergraph, on which we will apply their result.

[2] This is the same notion as the constraint-variable graph considered in various works on lower bounds for CSPs.

▶ **Definition 30.** Let $H = (V(H), E(H))$ be a k-uniform hypergraph. We define its incidence graph as the bipartite graph G_H defined on vertex sets $V(H)$ and $E(H)$, and edge set $\mathcal{E}$ defined as

$$\mathcal{E} \;:=\; \{(v,e) \;\mid\; v \in V(H),\; e \in E(H),\; v \in e\}\;.$$

Note that for any $u, v \in V(H)$, we have $d_{G_H}(u,v) = 2 \cdot d_H(u,v)$. We prove that for a locally sparse hypergraph H, its incidence graph G_H is also locally sparse.

▶ **Lemma 31.** *Let H be a k-uniform (τ,η)-sparse hypergraph on N vertices with $m = \gamma \cdot n$ hyperedges. Then the incidence graph G_H is (τ',η') sparse for $\tau' = \tau/k\cdot(1+\gamma)$ and $\eta' = \eta/(1+\eta)$.*

Proof. Let $\tau' = \tau/k\cdot(1+\gamma)$ and let G_H be the incidence graph with $N + m = (1+\gamma)\cdot N$ vertices. Let G' be is the densest subgraph of G_H, among all subgraphs of size at most $\tau' \cdot (N+m)$. Let the vertex set of G' be $V' \cup E'$ where $V' \subseteq V(H)$ and $E' \subseteq E(H)$, and let the edge-set be $\mathcal{E}'$. There cannot be any isolated vertices in G' since removing those will only increase the density.

Let $S \subseteq V(H)$ be the set of all vertices contained in all edges in E' i.e., $S := \{v \in V(H) \mid \exists e \in E' \text{ s.t. } v \in e\}$. Note that $V' \subseteq S$, since there are no isolated vertices, and $E' \subseteq E(S)$, where $E(S)$ denotes the set of hyperedges contained in S.

By our choice of parameters, $|S| \le k \cdot |E'| \le k \cdot \tau' \cdot (N+m) \le \tau \cdot N$. Thus, using the sparsity of H, we have

$$|E'| \;\le\; |E(S)| \;\le\; \frac{|S|}{k-1-\eta}\,.$$

Also, since each hyperedge of E' can include at most k vertices in S, and since each edge in $\mathcal{E}'$ is incident on a vertex in V', we have

$$|S| - |V'| \;\le\; k\cdot|E'| - |\mathcal{E}'|\;.$$

Combining the two inequalities gives

$$(k-1-\eta)\cdot|E'| \;\le\; |V'| + k\cdot|E'| - |\mathcal{E}'| \quad\Longrightarrow\quad |\mathcal{E}'| \;\le\; (1+\eta)\cdot|E'| + |V'|\;.$$

Hence, we get that $|\mathcal{E}'| \le \frac{|V'|+|E'|}{(1-\eta')}$ for $\eta' = \frac{\eta}{(1+\eta)}$. ◀

Charikar et al. [10] defined the following structural property of a graph.

▶ **Definition 32** ([10])**.** A graph G is ℓ-path decomposable if every 2-connected subgraph G' of G, such that G' is not an edge, contains a path of length ℓ such that every vertex of the path has degree at most 2 in G'.

The above property was also implicitly used by Arora et al. ([3]), who proved the following (see Lemma 2.12 in [3]):

▶ **Lemma 33.** *Let $\ell > 0$ be an integer and $0 < \eta < \frac{1}{3\ell-1} < 1$. Let G be a η-sparse graph with girth $\mathsf{g} > \ell$. Then G is ℓ-path decomposable.*

Recall that we defined the metrics d_μ and ρ_μ on H as (for $u \neq v$) :

$$d_\mu^H(u,v) \;:=\; 1 - (1-\mu)^{2\cdot d_H(u,v)} \qquad \text{and} \qquad \rho_\mu^H(u,v) \;:=\; \sqrt{\frac{2\cdot d_\mu^H(u,v) + \mu}{1+\mu}}\,,$$

For a graph G, we define the following two metrics, for $u \neq v$:

$$d_\mu^G(u,v) \;\coloneqq\; 1-(-1)^{d_G(u,v)}(1-\mu)^{d_G(u,v)} \quad \text{and} \quad \rho_\mu^G(u,v) \;\coloneqq\; \sqrt{\frac{2\cdot d_\mu^G(u,v)+\mu}{1+\mu}}\,.$$

We note that if H is a hypergraph and G_H is its incidence graph, then the metrics $d_\mu^{G_H}$ and $\rho_\mu^{G_H}$ restricted to $V(H)$, coincide with the metrics d_μ and ρ_μ defined on H. Charikar et al. proved the following theorem (see Theorem 5.2) in [12].

▶ **Theorem 34** ([12]). *Let G be a graph on n' vertices with maximum degree D. Let $t < \sqrt{n'}$ and $\ell > 0$ be such that for $t' = D^{\ell+1}\cdot t$, every subgraph of G on at most t' vertices is ℓ-path decomposable. Also, let μ, t and ℓ satisfy the relation $(1-\mu)^{\ell/9} \leq \frac{\mu}{2(t+1)}$. Then for every subset S of at most t vertices there exists a mapping ψ_S from S to* unit sphere *in ℓ_2 such that all $u, v \in S$:*

$$\|\psi_S(u)-\psi_S(v)\|_2 \;=\; \rho_\mu^G(u,v)\,.$$

We use this theorem to prove the main theorem of the section.

Proof of Theorem 9. Let $H' \sim \mathcal{H}_k(m,n,n_0,\Gamma)$ with $m=\gamma\cdot n$ hyperedges and $N = n_0\cdot n$ vertices. Given $\varepsilon > 0$, from Lemma 25 we have that with high probability at least $1-\varepsilon/2$, there exists H_1 such that the maximum degree of H_1 is at most $D = 100\cdot\log\left(\frac{2n_0}{\varepsilon}\right)\cdot k\cdot\gamma$ with $|\mathrm{E}(H')\setminus\mathrm{E}(H_1)| \leq (\varepsilon/2)\cdot m$.

Using Corollary 27 we also have that there exists $\delta > 0$, such that with probability at least $1-\varepsilon/4$ (for large enough n) H' has a sub-hypergraph H_2 with $\mathsf{g} \geq \delta\cdot\log n$ and $|\mathrm{E}(H')\setminus\mathrm{E}(H_2)| \leq (\varepsilon/4)\cdot m$. By Lemma 29, there exists $\eta = \Omega_{n_0,k,\gamma,\varepsilon}(1/(\log n))$ such that H' is (τ,η)-sparse with probability at least $1-\varepsilon/4$, for $\tau \geq n^{-1/4}$.

Hence with probability $1-\varepsilon$, we have that $H = (V(H'), E(H_1)\cap E(H_2))$ satisfies:

- Degree of H is bounded above by D.
- H is (τ,η)-sparse (for $\tau \geq n^{-1/4}$ and $\eta = \Omega_{n_0,k,\gamma,\varepsilon}(1/(\log n))$.
- Girth of H is at least $\mathsf{g} > \delta\cdot\log n$.
- $|\mathrm{E}(H')\setminus\mathrm{E}(H)| \leq \varepsilon\cdot m$.

We now show that the metric ρ_μ^H is locally ℓ_2 embeddable.

Let $G = G_H$ be the incidence graph for the hypergraph H. Note that $N \leq |\mathrm{V}(G)| \leq N\cdot(1+\gamma)$ and degree of G is also bounded by D. Since a cycle in G is also a cycle in H, the girth of G is at also least $\mathsf{g} \geq \delta\cdot\log n$.

By Lemma 31, we have G is $(\frac{\tau}{k(1+\gamma)}, \frac{\eta}{1+\eta})$-sparse. By Lemma 33, any subgraph of G on at most $\frac{\tau}{k(1+\gamma)}\cdot(N+m)$ vertices is ℓ-path decomposable for any $\ell \leq \min\{\mathsf{g}, 1/(4\eta)\}$. Since $D = 100\cdot k\gamma\cdot\log(2n_0/\varepsilon)$, there exists $\ell_0 = \Omega_{k,\gamma,n_0,\varepsilon}(\log n)$ such that $D^{\ell_0+1} \leq n^{1/6}$. We choose $\ell = \min\{\mathsf{g}, 1/(4\eta), \ell_0\}$.

Let μ_0 be the smallest μ such that $\exp(-\mu\ell/9) \leq \frac{\mu}{2(t+1)}$ (note that $\frac{1}{\mu}\cdot\exp(-\mu\ell/9)$ is decreasing in μ). Since we must have $\mu \geq 1/\ell$, there exists a μ_0 satisfying

$$\mu_0 \;\leq\; \frac{9}{\ell}\cdot(\ln(2(t+1))+\ln\ell)\;.$$

From our choice of ℓ, there exist constants $c = c(k,\gamma,n_0,\varepsilon)$ and $\theta = \theta(k,\gamma,n_0,\varepsilon) < 1/2$ such that $\mu_0 \leq c\cdot\frac{\log t+\log\log n}{\log n} < 1$ when $t \leq n^\theta$. Then, for any $\mu \in [\mu_0, 1)$, we have $(1-\mu)^{\ell/9} \leq \exp(-\mu\ell/9) \leq \frac{\mu}{2(t+1)}$.

We can now apply Theorem 34 to construct the embedding. Given any subset S of $\mathrm{V}(H)$ of size at most $t \leq n^\theta$, note that S is also a subset of $\mathrm{V}(G)$. Moreover, we have

$t \le n^\theta \le (N+m)^{1/2}$. Also, we have $t \cdot D^{\ell+1} \le n^{1/2} \cdot n^{1/6} = n^{2/3} \le \frac{\tau}{k(\gamma+1)} \cdot (N+m)$. Thus, any subgraph of G on $t \cdot D^{\ell+1}$ vertices is ℓ-path decomposable.

Thus, when $\mu \ge \mu_0$, by Theorem 34 there exists a mapping ψ_S from S to the unit sphere, such that for all $u, v \in S$, we have

$$\|\psi_S(u) - \psi_S(v)\|_2 \;=\; \rho_\mu^G(u,v) \;=\; \rho_\mu^H(u,v)\,,$$

where the last equality uses the fact that for all $u, v \in \mathrm{V}(H)$, $\rho_\mu^H(u,v) = \rho_\mu^G(u,v)$ since $d_G(u,v) = 2 \cdot d_H(u,v)$. ◀

B Omitted proofs Section 5

▶ **Claim 23.** *Let $\zeta \in \mathcal{C}_\delta(f)$ and let ν be the corresponding distribution supported in $f^{-1}(1)$ such that for all $i \in [k]$, we have $\zeta_i = \mathbb{E}_{\alpha\sim\nu}[(-1)^{\alpha_i}]$. Let $t_1, \ldots, t_k \in [0,1]$ be such that for all $i \in [k]$, $|t_i - |\zeta_i|| \le \varepsilon$ for $\varepsilon < \delta/2$. Then there exists a distribution ν' on $\{0,1\}^k$ such that*

$$\|\nu - \nu'\|_1 \;=\; O(k \cdot (\varepsilon/\delta)) \qquad \textit{and} \qquad \forall i \in [k], \;\; \mathop{\mathbb{E}}_{\alpha\sim\nu'}[(-1)^{\alpha_i}] \;=\; \mathrm{sign}(\zeta_i) \cdot t_i\,.$$

Proof. Let $r_j = \mathrm{sign}(\zeta_j) \cdot t_j$ be the desired bias of the j^{th} coordinate. Then, $|\zeta(j) - r_j| \le \varepsilon$ for all $j \in [k]$ We construct a sequence of distributions $\nu_0, \ldots, \nu_k$ such that $\nu_0 = \nu$ and $\nu_k = \nu'$. In $\bar{\nu}_j$, the biases are $(r_1, \ldots, r_j, \zeta_{j+1}, \ldots, \zeta_k)$.

The biases in ν_0 satisfy the above by definition. We obtain $\bar{\nu}_j$ from $\bar{\nu}_{j-1}$ as,

$$\nu_j = (1-\tau_j) \cdot \nu_{j-1} + \tau_j \cdot D_j\,,$$

where D_j is the distribution in which all bits, except for the j^{th} one, are set independently according to their biases in $\bar{\nu}_{j-1}$. For the j^{th} bit, we set it to $\mathrm{sign}(r_j - \zeta_j)$ (if $r_j - \zeta(j) = 0$, we can simply proceed with $\bar{\nu}_j = \bar{\nu}_{j-1}$). The biases for all except for the j^{th} bit are unchanged. For the j^{th} bit, the bias now becomes r_j if

$$r_j = (1-\tau_j) \cdot \zeta_j + \tau_j \cdot \mathrm{sign}(r_j - \zeta_j) \implies \tau_j \cdot (\mathrm{sign}(r_j - \zeta_j) - r_j) = (1-\tau_j) \cdot (r_j - \zeta_j)\,.$$

Since $\zeta \in \mathcal{C}_\delta(f)$, we know that $|\mathrm{sign}(r_j - \zeta(j)) - r_j| \ge \delta/2$. Also, $|r_j - \zeta(j))| \le \varepsilon$ by assumption. Thus, we can choose $\tau_j = O(\varepsilon/\delta)$ which gives that $\|\bar{\nu}_j - \bar{\nu}_{j-1}\|_1 = O(\varepsilon/\delta)$. The final bound then follows by triangle inequality. ◀

Query-to-Communication Lifting for P^{NP}*†

Mika Göös[1], Pritish Kamath[2], Toniann Pitassi[3], and Thomas Watson[4]

1 **Harvard University, Cambridge, MA, USA**
mika@seas.harvard.edu
2 **Massachusetts Institute of Technology, Cambridge, MA, USA**
pritish@mit.edu
3 **University of Toronto, Toronto, ON, Canada**
toni@cs.toronto.edu
4 **University of Memphis, Memphis, TN, USA**
Thomas.Watson@memphis.edu

Abstract

We prove that the P^{NP}-type query complexity (alternatively, decision list width) of any boolean function f is quadratically related to the P^{NP}-type communication complexity of a lifted version of f. As an application, we show that a certain "product" lower bound method of Impagliazzo and Williams (CCC 2010) fails to capture P^{NP} communication complexity up to polynomial factors, which answers a question of Papakonstantinou, Scheder, and Song (CCC 2014).

1998 ACM Subject Classification F.1.3 Complexity Measures and Classes

Keywords and phrases Communication Complexity, Query Complexity, Lifting Theorem, P^{NP}

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.12

1 Introduction

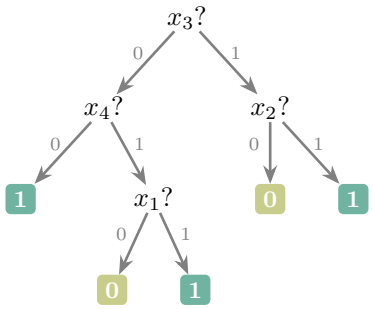
Broadly speaking, a **query-to-communication lifting theorem** (a.k.a. communication-to-query simulation theorem) translates, in a black-box fashion, lower bounds on some type of *query complexity* (a.k.a. decision tree complexity) [38, 6, 19] of a boolean function $f\colon \{0,1\}^n \to \{0,1\}$ into lower bounds on a corresponding type of *communication complexity* [23, 19, 27] of a two-party version of f. Table 1 lists several known results in this vein.

In this work, we provide a lifting theorem for P^{NP}-type query/communication complexity.

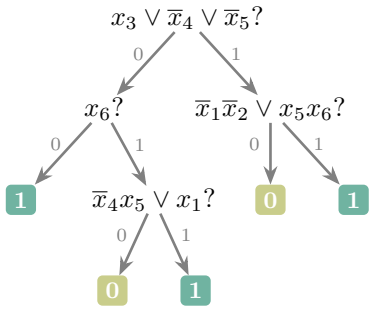
P^{NP} decision trees. Recall that a deterministic (i.e., P-type) decision tree computes an n-bit boolean function f by repeatedly querying, at unit cost, individual bits $x_i \in \{0,1\}$ of the input x until the value $f(x)$ is output at a leaf of the tree. A P^{NP} decision tree is more powerful: in each step, it can query/evaluate a width-k DNF of its choice, at the cost of k. Here k is simply the nondeterministic (i.e., NP-type) decision tree complexity of the predicate being evaluated at a node. The overall cost of a P^{NP} decision tree is the maximum over all inputs x of the sum of the costs of the individual queries that are made on input x. The P^{NP} query complexity of f, denoted $\mathsf{P}^{\mathsf{NPdt}}(f)$, is the least cost of a P^{NP} decision tree that computes f.

* A full version of the paper is available at https://eccc.weizmann.ac.il/report/2017/024/.

† P. Kamath was funded in parts by NSF grants CCF-1420956, CCF-1420692, CCF-1218547 and CCF-1650733. T. Watson was funded by NSF grant CCF-1657377.



Deterministic decision tree of cost 3



P^{NP} *decision tree of cost 4*

▶ **Example 1.** Consider the fabled *odd-max-bit* function [3, 7, 33, 36, 8] defined by $\textsc{Omb}(x) \coloneqq 1$ iff $x \neq 0^n$ and the largest index $i \in [n]$ such that $x_i = 1$ is odd. This function admits an efficient $O(\log n)$-cost P^{NP} decision tree: we can *find* the largest i with $x_i = 1$ by using a binary search that queries 1-DNFs of the form $\bigvee_{a \le j \le n} x_j$ for different $a \in [n]$.

P^{NP} communication protocols. Let $F\colon \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ be a two-party function, i.e., Alice holds $x \in \mathcal{X}$, Bob holds $y \in \mathcal{Y}$. A deterministic communication protocol can be viewed as a decision tree where in each step, at unit cost, it evaluates either an arbitrary predicate of Alice's input x or an arbitrary predicate of Bob's input y. A P^{NP} communication protocol [2, 15] is more powerful: in each step, it can evaluate an arbitrary predicate of the form $(x,y) \in \bigcup_{i \in [2^k]} R_i$ ("oracle query") at the cost of k (we always assume $k \ge 1$). Here each R_i is a rectangle (i.e., $R_i = X_i \times Y_i$ for some $X_i \subseteq \mathcal{X}$, $Y_i \subseteq \mathcal{Y}$) and k is just the usual nondeterministic communication complexity of the predicate being evaluated. The overall cost of a P^{NP} protocol is the maximum over all inputs (x,y) of the sum of the costs of the individual oracle queries that are made on input (x,y). The P^{NP} communication complexity of F, denoted $\mathsf{P}^{\mathsf{NPcc}}(F)$, is the least cost of a P^{NP} protocol computing F.

Note that if $F\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ can be written as a k-DNF on $2n$ variables, then the nondeterministic communication complexity of F, denoted $\mathsf{NP}^{\mathsf{cc}}(F)$, is at most $O(k \log n)$ bits: we can guess one of the $\le 2^k \binom{n}{k}$ many terms in the k-DNF and verify that the term evaluates to true. Consequently, any P^{NP} decision tree for a function f can be simulated efficiently by a P^{NP} protocol, regardless of how the input bits of f are split between Alice and Bob. That is, letting F be f equipped with any bipartition of the input bits, we have

$$\mathsf{P}^{\mathsf{NPcc}}(F) \;\le\; \mathsf{P}^{\mathsf{NPdt}}(f) \cdot O(\log n). \tag{1}$$

1.1 Main result

Our main result establishes a rough converse to inequality (1) for a special class of *composed*, or *lifted*, functions. For an n-bit function f and a two-party function $g\colon \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ (called a *gadget*), their composition $F \coloneqq f \circ g^n\colon \mathcal{X}^n \times \mathcal{Y}^n \to \{0,1\}$ is given by $F(x,y) \coloneqq f(g(x_1,y_1),\dots,g(x_n,y_n))$. We use as a gadget the popular *index* function $\textsc{Ind}_m\colon [m] \times \{0,1\}^m$ defined by $\textsc{Ind}_m(x,y) \coloneqq y_x$.

▶ **Theorem 2** (Lifting for P^{NP}). *Let $m = m(n) \coloneqq n^4$. For every $f\colon \{0,1\}^n \to \{0,1\}$,*

$$\mathsf{P}^{\mathsf{NPcc}}(f \circ \textsc{Ind}_m^n) \;\ge\; \sqrt{\mathsf{P}^{\mathsf{NPdt}}(f) \cdot \Omega(\log n)}.$$

Table 1 Some query-to-communication lifting theorems. The first four are formulated in the language of boolean functions (as in this paper); the last two are formulated in the language of combinatorial optimization.

Query model	Communication model	References
deterministic	deterministic	[28, 14, 10, 17]
nondeterministic	nondeterministic	[13, 11]
polynomial degree	rank	[35, 34, 29, 31]
conical junta degree	nonnegative rank	[13, 22]
Sherali–Adams	LP extension complexity	[9, 22]
sum-of-squares	SDP extension complexity	[24]

The lower bound is tight up to the square root, since (1) can be adapted for composed functions to yield $\mathsf{P}^{\mathsf{NPcc}}(f \circ \textsc{Ind}_m^n) \leq \mathsf{P}^{\mathsf{NPdt}}(f) \cdot O(\log m + \log n)$. The reason we incur a quadratic loss is because we actually prove a *lossless* lifting theorem for a related complexity measure that is known to capture P^{NP} query/communication complexity up to a quadratic factor, namely *decision lists*, discussed shortly in subsection 1.3.

1.2 Application

Impagliazzo and Williams [18] gave the following criteria – we call it the *product method* – for a function F to have large P^{NP} communication complexity. Here, a *product* distribution μ over $\mathcal{X} \times \mathcal{Y}$ is such that $\mu(x, y) = \mu_{\mathcal{X}}(x) \cdot \mu_{\mathcal{Y}}(y)$ for some distributions $\mu_{\mathcal{X}}, \mu_{\mathcal{Y}}$. A rectangle $R \subseteq \mathcal{X} \times \mathcal{Y}$ is *monochromatic* (relative to F) if F is constant on R.

> **Product method [18]:** *Let $F\colon \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ and suppose μ is a product distribution over $\mathcal{X} \times \mathcal{Y}$ such that $\mu(R) \leq \delta$ for every monochromatic rectangle R. Then*
>
> $$\mathsf{P}^{\mathsf{NPcc}}(F) \;\geq\; \Omega(\log(1/\delta)).$$

This should be compared with the well-known *rectangle size method* [20], [23, §2.4] (μ over $F^{-1}(1)$ such that $\mu(R) \leq \delta$ for all monochromatic R implies $\mathsf{NP}^{\mathsf{cc}}(F) \geq \Omega(\log(1/\delta))$), which is known to characterize nondeterministic communication complexity up to an additive $\Theta(\log n)$ term.

Papakonstantinou, Scheder, and Song [25, Open Problem 1] asked whether the product method can yield a tight P^{NP} communication lower bound for every function. This is especially relevant in light of the fact that all existing lower bounds against $\mathsf{P}^{\mathsf{NPcc}}$ (proved in [18, 25]) have used the product method (except those lower bounds that hold against an even stronger model: unbounded error randomized communication complexity, $\mathsf{UPP}^{\mathsf{cc}}$ [26]). We show that the product method can fail exponentially badly, even for total functions.

▶ **Theorem 3.** *There exists a total $F\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ satisfying the following.*

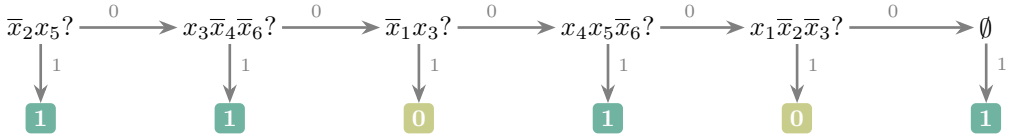
- *F has large P^{NP} communication complexity: $\mathsf{P}^{\mathsf{NPcc}}(F) \geq n^{\Omega(1)}$.*
- *For any product distribution μ over $\{0,1\}^n \times \{0,1\}^n$, there exists a monochromatic rectangle R that is large: $\log(1/\mu(R)) \leq \log^{O(1)} n$.*

1.3 Decision lists (DLs)

Conjunction DLs. The following definition is due to Rivest [30]: a *conjunction decision list* of width k is a sequence $(C_1, \ell_1), \ldots, (C_L, \ell_L)$ where each C_i is a conjunction of $\leq k$ literals

and $\ell_i \in \{0,1\}$ is a label. We assume for convenience that C_L is the empty conjunction (accepting every input). Given an input x, the conjunction decision list finds the least $i \in [L]$ such that $C_i(x) = 1$ and outputs ℓ_i. We define the conjunction decision list width of f, denoted $\mathsf{DL}^{\mathsf{dt}}(f)$, as the minimum k such that f can be computed by a width-k conjunction decision list. For example, $\mathsf{DL}^{\mathsf{dt}}(\textsc{Omb}) = 1$. This complexity measure is quadratically related to P^{NP} query complexity (for details, see full version of this paper [12]).

▶ **Fact 4.** *For all* $f\colon \{0,1\}^n \to \{0,1\}$, $\Omega(\mathsf{DL}^{\mathsf{dt}}(f)) \le \mathsf{P}^{\mathsf{NPdt}}(f) \le O(\mathsf{DL}^{\mathsf{dt}}(f)^2 \cdot \log n)$.



A conjunction decision list of width 3

Rectangle DLs. A communication complexity variant of decision lists was introduced by Papakonstantinou, Scheder, and Song [25] (they called them *rectangle overlays*). A *rectangle decision list* of cost k is a sequence $(R_1, \ell_1), \dots, (R_{2^k}, \ell_{2^k})$ where each R_i is a rectangle and $\ell_i \in \{0,1\}$ is a label. We assume for convenience that R_{2^k} contains every input. Given an input (x,y), the rectangle decision list finds the least $i \in [2^k]$ such that $(x,y) \in R_i$ and outputs ℓ_i. We define the rectangle decision list complexity of F, denoted $\mathsf{DL}^{\mathsf{cc}}(F)$, as the minimum k such that F can be computed by a cost-k rectangle decision list. We again have a quadratic relationship [25, Theorem 3] (for details, see full version of this paper [12]).

▶ **Fact 5.** *For all* $F\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$, $\Omega(\mathsf{DL}^{\mathsf{cc}}(F)) \le \mathsf{P}^{\mathsf{NPcc}}(F) \le O(\mathsf{DL}^{\mathsf{cc}}(F)^2)$.

DLs are combinatorially slightly more comfortable to work with than P^{NP} decision trees/protocols. This is why our main lifting theorem (Theorem 2) is in fact derived as a corollary of a *lossless* lifting theorem for DLs.

▶ **Theorem 6** (Lifting for DL). *Let* $m = m(n) \coloneqq n^4$. *For every* $f\colon \{0,1\}^n \to \{0,1\}$,

$$\mathsf{DL}^{\mathsf{cc}}(f \circ \textsc{Ind}_m^n) \;=\; \mathsf{DL}^{\mathsf{dt}}(f) \cdot \Theta(\log n).$$

Indeed, Theorem 2 follows because $\mathsf{P}^{\mathsf{NPcc}}(f \circ \textsc{Ind}_m^n) \ge \Omega(\mathsf{DL}^{\mathsf{cc}}(f \circ \textsc{Ind}_m^n)) \ge \Omega(\mathsf{DL}^{\mathsf{dt}}(f) \cdot \log n) \ge \Omega((\mathsf{P}^{\mathsf{NPdt}}(f)/\log n)^{1/2} \cdot \log n) = (\mathsf{P}^{\mathsf{NPdt}}(f) \cdot \Omega(\log n))^{1/2}$, where the first inequality is by Fact 5, the second is by Theorem 6, and the third is by Fact 4. We mention that Theorems 2 and 6, as well as Facts 4 and 5, in fact hold for all partial functions.

As a curious aside, we mention that a time-bounded analogue of decision lists (capturing P^{NP}) has also been studied in a work of Williams [39].

1.4 Separation between P^{NP} and DL

Facts 4 and 5 show that decision lists can be converted to P^{NP} decision trees/protocols with a quadratic overhead. Is this conversion optimal? In other words, are there functions that witness a quadratic gap between P^{NP} and DL? We at least show that *if a lossless lifting theorem holds for* P^{NP}, then such a quadratic gap indeed exists for communication complexity.

▶ **Conjecture 7.** *There is an* $m = m(n) \coloneqq n^{\Theta(1)}$ *such that for every* $f\colon \{0,1\}^n \to \{0,1\}$,

$$\mathsf{P}^{\mathsf{NPcc}}(f \circ \textsc{Ind}_m^n) \;=\; \mathsf{P}^{\mathsf{NPdt}}(f) \cdot \Theta(\log n).$$

Our bonus contribution here (proof deferred to the full version [12]) shows that the simple $O(\log n)$-cost P^{NP} decision tree for the odd-max-bit function is optimal:

▶ **Theorem 8.** $\mathsf{P}^{\mathsf{NPdt}}(\textsc{Omb}) \geq \Omega(\log n)$.

▶ **Corollary 9.** *The second inequality of Fact 4 is tight (i.e.,* $\mathsf{P}^{\mathsf{NPdt}}(f) \geq \Omega(\mathsf{DL}^{\mathsf{dt}}(f)^2 \cdot \log n)$ *for some* f*), and assuming Conjecture 7, the second inequality of Fact 5 is tight (i.e.,* $\mathsf{P}^{\mathsf{NPcc}}(F) \geq \Omega(\mathsf{DL}^{\mathsf{cc}}(F)^2)$ *for some* F*).*

This corollary is witnessed by $f := \textsc{Omb}$ (which has $\mathsf{DL}^{\mathsf{dt}}(f) \leq O(1)$ and $\mathsf{P}^{\mathsf{NPdt}}(f) \geq \Omega(\log n)$) and its lifted version $F := \textsc{Omb} \circ \textsc{Ind}_m^n$ (which has $\mathsf{DL}^{\mathsf{cc}}(F) \leq O(\log n)$ and $\mathsf{P}^{\mathsf{NPcc}}(F) \geq \Omega(\log^2 n)$ under Conjecture 7). One caveat is that we have only shown the corollary for an extreme setting of parameters (constant $\mathsf{DL}^{\mathsf{dt}}(f)$ and logarithmic $\mathsf{DL}^{\mathsf{cc}}(F)$). It would be interesting to show a separation for functions of $n^{\Omega(1)}$ decision list complexity.

2 Preliminaries: Decision List Lower Bound Techniques

We present two basic lemmas in this section that allow one to prove lower bounds on conjunction/rectangle decision lists. First we recall the proof of the product method, which will be important for us, as we will extend the proof technique in both section 3 and section 4.

▶ **Lemma 10** (Product method for $\mathsf{DL}^{\mathsf{cc}}$)**.** *Let* $F\colon \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ *and suppose* μ *is a product distribution over* $\mathcal{X} \times \mathcal{Y}$*. Then* F *admits a monochromatic rectangle* R *with* $\log(1/\mu(R)) \leq O(\mathsf{DL}^{\mathsf{cc}}(F))$.

Proof (from [18, 25]). Let $(R_1, \ell_1), \ldots, (R_{2^k}, \ell_{2^k})$ be an optimal rectangle decision list of cost $k := \mathsf{DL}^{\mathsf{cc}}(F)$ computing F. Recall we assume that $R_{2^k} = \mathcal{X} \times \mathcal{Y}$ contains every input. We find a monochromatic R with $\mu(R) \geq 2^{-2k}$ via the following process.

We initialize $X := \mathcal{X}$ and $Y := \mathcal{Y}$ and iterate the following for $i = 1, \ldots, 2^k$ rounds, shrinking the rectangle $X \times Y$ in each round.

(†) *Round* i*:* (loop invariant: $R_i \cap X \times Y$ is a monochromatic rectangle)

Write $R_i \cap X \times Y = X_i \times Y_i$ and test whether $\mu(X_i \times Y_i) = \mu_{\mathcal{X}}(X_i) \cdot \mu_{\mathcal{Y}}(Y_i)$ is at least 2^{-2k}. Suppose not, as otherwise we are successful. Then either $\mu_{\mathcal{X}}(X_i) < 2^{-k}$ or $\mu_{\mathcal{Y}}(Y_i) < 2^{-k}$; say the former. We now "delete" the rows X_i from consideration by updating $X \leftarrow X \setminus X_i$.

Note that since $R_i \cap X \times Y$ is removed from $X \times Y$ in each unsuccessful round, it must hold (inductively) that $\bigcup_{j<i} R_j$ is disjoint from $X \times Y$ at the start of the i-th round, and so $R_i \cap X \times Y$ is indeed monochromatic (since it only contains points for which R_i is the first rectangle in the decision list to contain them, which means F evaluates to ℓ_i). The process starts out with $\mu(X \times Y) = 1$ and in each unsuccessful round the quantity $\mu(X \times Y)$ decreases by $< 2^{-k}$. Some round must succeed, as otherwise the process would finish with $X \times Y = \emptyset$ and hence $\mu(X \times Y) = 0$ in 2^k rounds, which is impossible. ◀

Recall that our Theorem 3 states that the product method is not complete for the measure $\mathsf{DL}^{\mathsf{cc}}$. By contrast, we are able to give an alternative characterization for the analogous query complexity measure $\mathsf{DL}^{\mathsf{dt}}$. We do not know if this characterization has been observed in the literature before.

▶ **Lemma 11** (Characterization for $\mathsf{DL}^{\mathsf{dt}}$)**.** *Let* $f\colon \{0,1\}^n \to \{0,1\}$*. Then* $\mathsf{DL}^{\mathsf{dt}}(f) \leq k$ *iff for every nonempty* $Z \subseteq \{0,1\}^n$ *there exists an* $\ell \in \{0,1\}$ *and a width-*k *conjunction that accepts an input in* $Z_\ell := Z \cap f^{-1}(\ell)$ *but none in* $Z_{1-\ell}$*.*

Proof. Suppose f has a width-k conjunction decision list $(C_1, \ell_1), (C_2, \ell_2), \ldots, (C_L, \ell_L)$. The first C_i that accepts an input in Z (such an i must exist since the last C_L accepts every input) must accept an input in Z_{ℓ_i} but none in $Z_{1-\ell_i}$ (since all inputs in $C_i^{-1}(1) \cap Z$ are such that C_i is the first conjunction in the decision list to accept them).

Conversely, assume the right side of the "iff" holds. Then we can build a conjunction decision list for f iteratively as follows. Start with $Z = \{0,1\}^n$. Let C_1 be a width-k conjunction that accepts an input in some Z_{ℓ_1} but none in $Z_{1-\ell_1}$, and remove from Z all inputs accepted by C_1. Then continue with the new Z: let C_2 be a width-k conjunction that accepts an input in some Z_{ℓ_2} but none in $Z_{1-\ell_2}$, and further remove from Z all inputs accepted by C_2. Once Z becomes empty (this must happen since the right side of the iff holds for all nonempty Z), we have constructed a conjunction decision list $(C_1, \ell_1), (C_2, \ell_2), \ldots$ for f. ◀

3 Proof of the Lifting Theorem

In this section we prove Theorem 6, restated here for convenience.

▶ **Theorem 6** (Lifting for DL). *Let $m = m(n) := n^4$. For every $f\colon \{0,1\}^n \to \{0,1\}$,*

$$\mathsf{DL}^{\mathsf{cc}}(f \circ \textsc{Ind}_m^n) \;=\; \mathsf{DL}^{\mathsf{dt}}(f) \cdot \Theta(\log n).$$

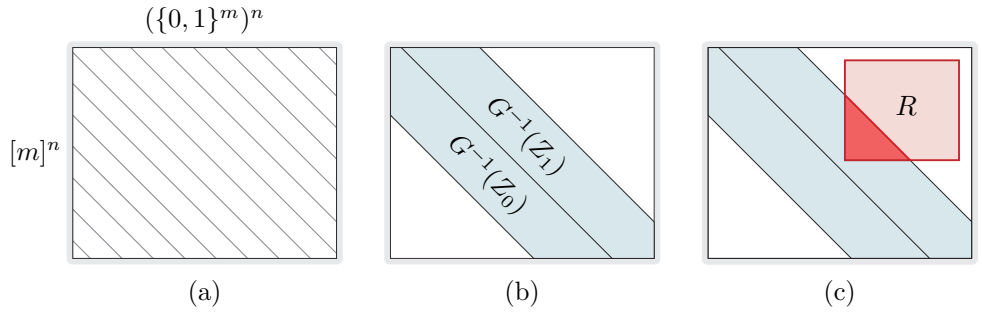
We use the abbreviations $g := \textsc{Ind}_m\colon [m] \times \{0,1\}^m \to \{0,1\}$ and $F := f \circ g^n$.

The upper bound of Theorem 6 is straightforward: given a width-k conjunction decision list for f (which necessarily has length $\le 2^k \binom{n}{k} \le n^{O(k)}$), we can form a rectangle decision list for F by transforming each labeled conjunction into a set of same-labeled rectangles (which can be ordered arbitrarily among themselves), one for each of the m^k ways of choosing a row from each of the copies of g corresponding to bits read by the conjunction – for a total of $n^{O(k)} \cdot m^k \le n^{O(k)}$ rectangles and hence a cost of $k \cdot O(\log n)$. For example, if $k = 2$ and the conjunction is $z_1 \overline{z}_2$, then for each $x_1, x_2 \in [m]$ there would be a rectangle consisting of all inputs with that value of x_1, x_2 and with y_1, y_2 such that $g(x_1, y_1) = 1$ and $g(x_2, y_2) = 0$. For the rest of this section, we prove the matching lower bound.

3.1 Overview

Fix an optimal rectangle decision list $(R_1, \ell_1), \ldots, (R_{2^k}, \ell_{2^k})$ for F. By our characterization of $\mathsf{DL}^{\mathsf{dt}}$ (Theorem 11) it suffices to show that for every nonempty $Z \subseteq \{0,1\}^n$ there is a width-$O(k/\log n)$ conjunction that accepts an input in $Z_\ell := Z \cap f^{-1}(\ell)$ for some $\ell \in \{0,1\}$, but none in $Z_{1-\ell}$. Thus fix some nonempty Z henceforth.

Write $G := g^n$ for short. We view the communication matrix of F as being partitioned into *slices* $G^{-1}(z) = \{(x,y) : G(x,y) = z\}$, one for each $z \in \{0,1\}^n$; see (a) below. We focus naturally on the slices corresponding to Z, namely $G^{-1}(Z) = \bigcup_{z \in Z} G^{-1}(z)$, which is further partitioned into $G^{-1}(Z_0)$ and $G^{-1}(Z_1)$; see (b) below. Our goal is to find a rectangle R that touches $G^{-1}(Z_\ell)$ (for some ℓ) but not $G^{-1}(Z_{1-\ell})$, and such that $G(R) = C^{-1}(1)$ for a width-$O(k/\log n)$ conjunction C; see (c) below. Thus $C^{-1}(1)$ touches Z_ℓ but not $Z_{1-\ell}$, as desired.



We find such an R as follows. We maintain a rectangle $X \times Y$, which is initially the whole domain of F and which we iteratively shrink. In each round, we consider the next rectangle R_i in the decision list, and one of two things happens. Either:

- The round is declared unsuccessful, in which case we remove from $X \times Y$ a small number of rows and columns that together cover all of $R_i \cap X \times Y \cap G^{-1}(Z)$. This guarantees that throughout the whole execution, by the i-th round, $\bigcup_{j<i}(R_j \cap G^{-1}(Z))$ has been removed from $X \times Y$ – thus every input in $R_i \cap X \times Y \cap G^{-1}(Z)$ is such that R_i is the first rectangle in the decision list that contains it, so it is in $G^{-1}(Z_{\ell_i}) \subseteq F^{-1}(\ell_i)$ by the definition of decision lists.

Or,

- Success is declared, in which case it will hold that $R_i \cap X \times Y$ touches $G^{-1}(Z)$ – in fact, it touches $G^{-1}(Z_{\ell_i})$ but not $G^{-1}(Z_{1-\ell_i})$, by the above – and we can restrict $R_i \cap X \times Y$ to a subrectangle R that still touches $G^{-1}(Z_{\ell_i})$ but is such that $G(R)$ is fixed on $O(k/\log n)$ coordinates and has full support on the remaining coordinates. In other words, $G(R) = C^{-1}(1)$ for a width-$O(k/\log n)$ conjunction C.

This process is a variation of the process (†) from the product method (Theorem 10). The difference is that the Z-slices, $G^{-1}(Z)$, now play the role of the product distribution, and we maintain the monochromatic property for $R_i \cap X \times Y$ only inside the Z-slices. Another difference is that in each unsuccessful round we remove *both* rows *and* columns from $X \times Y$ (not *either–or* as in (†)).

To flesh out this outline, we need to specify how to determine whether a round is successful, which rows and columns to remove if not, and how to restrict to the desired R if so, and we need to argue that the process will terminate with success.

3.2 Tools

We will need to find a rectangle R such that $G(R)$ is fixed on few coordinates and has full support on the remaining coordinates. We now describe some tools that help us achieve this. First of all, under what conditions on $R = A \times B$ can we guarantee that $G(R)$ has full support over all n coordinates?

▶ **Definition 12** (Blockwise-density [13]). *$A \subseteq [m]^n$ is called δ-dense* if the uniform random variable $\boldsymbol{x}$ over A satisfies the following: for every nonempty $I \subseteq [n]$, the blocks $\boldsymbol{x}_I$ have min-entropy rate at least δ, that is, $\mathbf{H}_\infty(\boldsymbol{x}_I) \geq \delta \cdot |I| \log m$. Here, $\boldsymbol{x}_I$ is marginally distributed over $[m]^I$, and $\mathbf{H}_\infty(\boldsymbol{x}) := \min_x \log(1/\mathbf{Pr}[\boldsymbol{x} = x])$ is the usual min-entropy of a random variable (see, e.g., Vadhan's monograph [37] for an introduction).

▶ **Definition 13** (Deficiency). For $B \subseteq (\{0,1\}^m)^n$, we define $\mathbf{D}_\infty(B) := mn - \log|B|$ (equivalently, $|B| = 2^{mn-\mathbf{D}_\infty(B)}$), representing the log-size deficiency of B compared to the

universe $(\{0,1\}^m)^n$. (The notation $\mathbf{D}_\infty$ was chosen partly because this corresponds to the Rényi max-divergence between the uniform distributions over B and over $(\{0,1\}^m)^n$.)

▶ **Lemma 14** (Full support). *If $A \subseteq [m]^n$ is 0.9-dense and $B \subseteq (\{0,1\}^m)^n$ satisfies $\mathbf{D}_\infty(B) \leq m^{0.3}$, then $G(A \times B) = \{0,1\}^n$ (i.e., for every $z \in \{0,1\}^n$ there are $x \in A$ and $y \in B$ with $G(x,y) = z$).*

We prove Theorem 14 in subsection 3.4 using the probabilistic method: we show for a suitably randomly chosen rectangle $U \times V \subseteq G^{-1}(z)$, (i) U intersects A with high probability, and (ii) V intersects B with high probability. The proof of (i) uses the second moment method (which is different from how blockwise-density was employed in previous work [13]). The proof of (ii) is a tightened analysis of a combination of arguments from [28, 14] (which were not stated in those papers with the high-probability guarantee we need). The latter papers proved the full support property under a different assumption on A, which they called "thickness".

Theorem 14 gives us the full support property assuming A is blockwise-dense and B has low deficiency. How can we get blockwise-density? Our tool for this is the following claim, which follows from [13]; we provide the simple argument.

▶ **Claim 15.** *If $A \subseteq [m]^n$ satisfies $|A| \geq m^n/2^{O(k)}$ then there exists an $I \subseteq [n]$ of size $|I| \leq O(k/\log n)$ and an $A' \subseteq A$ such that A' is fixed on I and 0.9-dense on $\overline{I} \coloneqq [n] \setminus I$.*

Proof. If A is 0.9-dense, then we can take $I = \emptyset$ and $A' = A$, so assume not. Letting $\boldsymbol{x}$ be the uniform random variable over A, take $I \subseteq [n]$ to be a maximal subset for which there is a violation of blockwise-density: $\mathbf{H}_\infty(\boldsymbol{x}_I) < 0.9 \cdot |I| \log m$. From $\mathbf{H}_\infty(\boldsymbol{x}) \geq n \log m - O(k)$ we deduce $\mathbf{H}_\infty(\boldsymbol{x}_I) \geq |I| \log m - O(k)$ since marginalizing out $|\overline{I}| \log m$ bits may only cause the min-entropy to go down by $|\overline{I}| \log m$. Combining these, we get $|I| \log m - O(k) < 0.9 \cdot |I| \log m$, so $|I| \leq O(k/\log n)$.

Let $\alpha \in [m]^I$ be an outcome for which $\mathbf{Pr}[\boldsymbol{x}_I = \alpha] > 2^{-0.9 \cdot |I| \log m}$, and take $A' \coloneqq \{x \in A : x_I = \alpha\}$, which is fixed on I. To see that A' is 0.9-dense on $\overline{I}$, let $\boldsymbol{x}'$ be the uniform random variable over A' and note that if $\mathbf{H}_\infty(\boldsymbol{x}'_J) < 0.9 \cdot |J| \log m$ for some nonempty $J \subseteq \overline{I}$, a straightforward calculation shows that then $\boldsymbol{x}_{I \cup J}$ would also have min-entropy rate < 0.9, contradicting the maximality of I. ◀

3.3 Finding R

We initialize $X \coloneqq [m]^n$ and $Y \coloneqq (\{0,1\}^m)^n$ and iterate the following for $i = 1, \ldots, 2^k$ rounds.

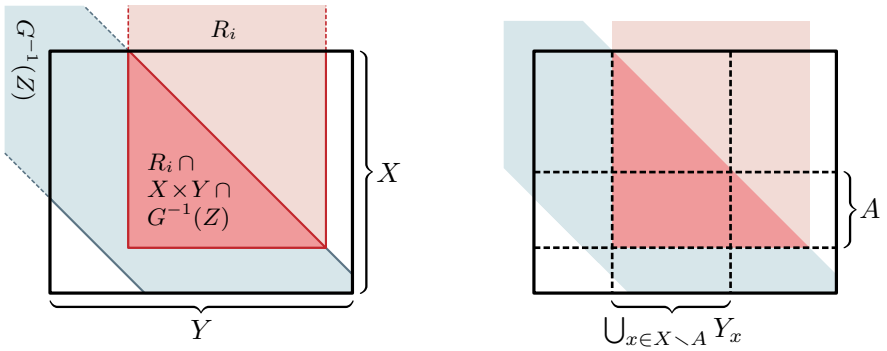
(‡) *Round i:* (loop invariant: $R_i \cap X \times Y \cap G^{-1}(Z)$ is monochromatic)

Define a set $A \subseteq X$ of *weighty rows* as

$$A \coloneqq \{x \in X : |Y_x| \geq 2^{mn - 3n \log m}\} \quad \text{where} \quad Y_x \coloneqq \{y \in Y : (x,y) \in R_i \cap G^{-1}(Z)\}.$$

Test whether there are many weighty rows: $|A| \geq m^n/2^{k+1}$?

- If *no*, we update $X \leftarrow X \setminus A$ and $Y \leftarrow Y \setminus \bigcup_{x \in X \setminus A} Y_x$ and proceed to the next round. Since $R_i \cap G^{-1}(Z)$ has been removed from $X \times Y$, this ensures our loop invariant, as explained in subsection 3.1.
- If *yes*, we declare this round a *success* and halt.



We shortly argue that the process halts with success. First, we show how to find a desired R assuming the process is successful in round i (with associated sets R_i, $X \times Y$, A, and Y_x for $x \in X$). Using Claim 15, obtain $A' \subseteq A$ which is fixed to α on some $I \subseteq [n]$ of size $O(k/\log n)$ and is 0.9-dense on $\overline{I}$. Pick any $x' \in A'$, and define $\gamma \in \{0,1\}^I$ to be a value that maximizes the size of $B := \{y \in Y_{x'} : g^I(\alpha, y_I) = \gamma\}$. Note that $|B| \geq |Y_{x'}|/2^{|I|} \geq 2^{mn-3n\log m - O(k/\log n)} \geq 2^{mn-m^{0.3}}$ since $x' \in A$ and $k \leq n\log(2m)$.

We claim that $R := A' \times B$ can serve as our desired rectangle. Certainly, R touches $G^{-1}(Z_{\ell_i})$ (at (x', y) for any $y \in B$) but not $G^{-1}(Z_{1-\ell_i})$ by the loop invariant (since $R \subseteq R_i \cap X \times Y$). Also, $G(R)$ is fixed to γ on I. Defining

$$A'_{\overline{I}} := \{x_{\overline{I}} \in [m]^{\overline{I}} : \alpha x_{\overline{I}} \in A'\} \qquad \text{and} \qquad B_{\overline{I}} := \{y_{\overline{I}} \in (\{0,1\}^m)^{\overline{I}} : \exists y_I \text{ s.t. } y_I y_{\overline{I}} \in B\}$$

to be the projections of A' and B to the coordinates $\overline{I}$, we have that

$$A'_{\overline{I}} \text{ is 0.9-dense} \qquad \text{and} \qquad \mathbf{D}_\infty(B_{\overline{I}}) \leq \mathbf{D}_\infty(B) \leq m^{0.3}$$

(noting that $\mathbf{D}_\infty(B_{\overline{I}})$ is relative to $(\{0,1\}^m)^{\overline{I}}$). Applying Theorem 14 to $A'_{\overline{I}} \times B_{\overline{I}}$ shows that $G(R)$ has full support on $\overline{I}$. In summary, "$z_I = \gamma$" is the conjunction we were looking for.

We now argue that the process halts with success. In each unsuccessful round, we remove $|A| < m^n/2^{k+1}$ rows from X and at most $\sum_{x \in X \setminus A} |Y_x| < m^n \cdot 2^{mn-3n\log m} \leq 2^{mn}/2^{k+1}$ columns from Y (since $k+1 \leq 2n\log m$). Suppose for contradiction that all 2^k rounds are unsuccessful; then at most half of the rows and half of the columns are removed altogether. Supposedly the set $X \times Y$ we finish with is disjoint from $\bigcup_{i \in [2^k]} (R_i \cap G^{-1}(Z)) = G^{-1}(Z)$. But since Z is nonempty, this contradicts the fact that $G(X \times Y)$ has full support by Theorem 14 (as it is straightforward to check that since $X \times Y$ contains at least half the rows and half the columns, it also satisfies the assumptions of the lemma).

This concludes the proof of Theorem 6, except for the proof of Theorem 14.

3.4 Full Support Lemma

▶ **Lemma 16** (Full support). *If $A \subseteq [m]^n$ is 0.9-dense and $B \subseteq (\{0,1\}^m)^n$ satisfies $\mathbf{D}_\infty(B) \leq m^{0.3}$, then $G(A \times B) = \{0,1\}^n$ (i.e., for every $z \in \{0,1\}^n$ there are $x \in A$ and $y \in B$ with $G(x,y) = z$).*

For coordinates $I \subseteq [n]$ we define $B_I := \{y_I \in (\{0,1\}^m)^I : \exists y_{\overline{I}} \text{ s.t. } y_I y_{\overline{I}} \in B\}$ as the *projection* of B onto I. Moreover, for $V \subseteq \{0,1\}^m$ and $i \in [n]$ we let $B^{i,V} := \{y \in B : y_i \in V\}$ be the *restriction* of the i-th coordinate to be in V. We will often use combinations of these notations; e.g., $B^{n,V}_{[n-1]}$ denotes the restriction of the n-th coordinate to be in V, subsequently projected on the coordinates in $[n-1]$.

We write random variables as bold letters. For a random variable $\boldsymbol{y}$ supported on B, $\boldsymbol{y}_I$ denotes the marginal distribution of $\boldsymbol{y}$ on the coordinates in I. In contrast, B_I only denotes the set obtained by projecting B on the coordinates in I, without any distribution associated to it. Note that while $\mathbf{D}_\infty(B)$ is the deficiency relative to $(\{0,1\}^m)^n$, the quantity $\mathbf{D}_\infty(B_I)$ is the deficiency relative to $(\{0,1\}^m)^I$; i.e., $\mathbf{D}_\infty(B_I) = m|I| - \log|B_I|$.

Theorem 14 follows from the following two claims.

▶ **Claim 17** (Alice side). *Suppose $A \subseteq [m]^n$ is 0.9-dense. Choose $\boldsymbol{U} \coloneqq \boldsymbol{U}_1 \times \cdots \times \boldsymbol{U}_n \subseteq [m]^n$ uniformly at random where each $\boldsymbol{U}_i \subseteq [m]$ is of size $|\boldsymbol{U}_i| = m^{0.36}$. Then*

$$\mathbf{Pr}[A \cap \boldsymbol{U} \neq \emptyset] \;\geq\; 1 - 2m^{-0.01}.$$

▶ **Claim 18** (Bob side). *Let $z \in \{0,1\}$ and suppose $B \subseteq (\{0,1\}^m)^n$ satisfies $\mathbf{D}_\infty(B) \leq m^{0.31}$. Choose $\boldsymbol{U} \subseteq [m]$, $|\boldsymbol{U}| = m^{0.36}$, uniformly at random and let $\boldsymbol{V} \coloneqq \{y \in \{0,1\}^m : \forall j \in \boldsymbol{U},\ y_j = z\}$. Then*

$$\begin{aligned} &\textit{for } n \geq 2\textit{:} && \mathbf{Pr}\left[\mathbf{D}_\infty\left(B^{n,\boldsymbol{V}}_{[n-1]}\right) \leq \mathbf{D}_\infty(B) + 1\right] &&\geq\; 1 - 60m^{-0.28},\\ &\textit{for } n = 1\textit{:} && \mathbf{Pr}\left[B \cap \boldsymbol{V} \neq \emptyset\right] &&\geq\; 1 - 60m^{-0.28}. \end{aligned}$$

We prove the Alice side claim shortly using the second moment method. The Bob side claim follows by a tightened analysis of arguments from [28, 14], which we provide in the full version of the paper [12]. Let us see why these two claims imply Theorem 14.

Proof of Theorem 14. Our goal is to show that for each $z \in \{0,1\}^n$ we have $A \times B \cap G^{-1}(z) \neq \emptyset$. Choose $\boldsymbol{U} \coloneqq \boldsymbol{U}_1 \times \cdots \times \boldsymbol{U}_n \subseteq [m]^n$, $|\boldsymbol{U}_i| = m^{0.36}$, uniformly at random. Correspondingly, define $\boldsymbol{V} \coloneqq \boldsymbol{V}_1 \times \cdots \times \boldsymbol{V}_n$ where $\boldsymbol{V}_i \coloneqq \{y \in \{0,1\}^m : \forall j \in \boldsymbol{U}_i,\ y_j = z_i\}$. We have $\boldsymbol{U} \times \boldsymbol{V} \subseteq G^{-1}(z)$ by construction so it suffices to show that $A \times B \cap \boldsymbol{U} \times \boldsymbol{V}$ is nonempty with positive probability. To this end, we show that the events $A \cap \boldsymbol{U} \neq \emptyset$ and $B \cap \boldsymbol{V} \neq \emptyset$ both happen with high probability, and hence, by a union bound, $A \times B \cap \boldsymbol{U} \times \boldsymbol{V}$ is nonempty with high probability. The Alice side claim (Claim 17) already shows $A \cap \boldsymbol{U} \neq \emptyset$ w.h.p., so it remains to consider $B \cap \boldsymbol{V}$.

Define $\boldsymbol{B}^{\triangleright i} \coloneqq B \cap (((\{0,1\}^m)^i \times \boldsymbol{V}_{i+1} \times \cdots \times \boldsymbol{V}_n)$. That is, $\boldsymbol{B}^{\triangleright i}$ is obtained by restricting the j-th coordinate to be in $\boldsymbol{V}_j$ for $i+1 \leq j \leq n$. Note that $\boldsymbol{B}^{\triangleright n} = B$, $\boldsymbol{B}^{\triangleright i-1} = (\boldsymbol{B}^{\triangleright i})^{i,\boldsymbol{V}_i}$ and $\boldsymbol{B}^{\triangleright 0} = B \cap \boldsymbol{V}$. Let $\widehat{\boldsymbol{B}}^{\triangleright i} \coloneqq \boldsymbol{B}^{\triangleright i}_{[i]}$ be the projection of $\boldsymbol{B}^{\triangleright i}$ onto $[i]$. We define the following events E_i:

$$\begin{aligned} &\text{for } i \geq 2\text{:} && E_i \iff \mathbf{D}_\infty(\widehat{\boldsymbol{B}}^{\triangleright i-1}) \leq \mathbf{D}_\infty(\widehat{\boldsymbol{B}}^{\triangleright i}) + 1,\\ &\text{for } i = 1\text{:} && E_1 \iff \widehat{\boldsymbol{B}}^{\triangleright 1} \cap \boldsymbol{V}_1 \neq \emptyset. \end{aligned}$$

Note that $\widehat{\boldsymbol{B}}^{\triangleright 1} \cap \boldsymbol{V}_1 \neq \emptyset$ implies that $\boldsymbol{B}^{\triangleright 0} = B \cap \boldsymbol{V} \neq \emptyset$. Conditioned on $E_n \cap \cdots \cap E_{i+1}$, we have

$$\mathbf{D}_\infty\left(\widehat{\boldsymbol{B}}^{\triangleright i}\right) \;\leq\; \mathbf{D}_\infty\left(\widehat{\boldsymbol{B}}^{\triangleright n}\right) + n - i - 1 \;\leq\; m^{0.3} + n \;\leq\; m^{0.31}$$

and thus for $i \geq 2$, we have from Claim 18 that $\mathbf{D}_\infty(\widehat{\boldsymbol{B}}^{\triangleright i-1}) \leq \mathbf{D}_\infty(\widehat{\boldsymbol{B}}^{\triangleright i}) + 1$ holds with probability at least $1 - 60m^{-0.28}$. Thus

$$\mathbf{Pr}[E_i \mid E_n \cap \cdots \cap E_{i+1}] \;\geq\; 1 - 60m^{-0.28}.$$

Also, conditioned on $E_n \cap \cdots \cap E_2$, we have $\mathbf{D}_\infty(\widehat{\boldsymbol{B}}^{\triangleright 1}) \leq m^{0.31}$, and hence using the case of $n = 1$ in Claim 18, $\mathbf{Pr}[\widehat{\boldsymbol{B}}^{\triangleright 1} \cap \boldsymbol{V}_1 \neq \emptyset] \geq 1 - 60m^{-0.28}$. That is,

$$\mathbf{Pr}[E_1 \mid E_n \cap \cdots \cap E_2] \;\geq\; 1 - 60m^{-0.28}.$$

Now we are able to show $B \cap \boldsymbol{V} \neq \emptyset$ w.h.p., which concludes the proof:

$$\begin{aligned}
\mathbf{Pr}[B \cap \boldsymbol{V} \neq \emptyset] &\geq \mathbf{Pr}[E_1] \\
&\geq \mathbf{Pr}[E_n \cap \cdots \cap E_1] \\
&= \textstyle\prod_{i=1}^{n} \mathbf{Pr}[E_i \mid E_n \cap \cdots \cap E_{i+1}] \\
&\geq (1 - 60m^{-0.28})^n \\
&\geq 1 - 60nm^{-0.28} \\
&= 1 - 60m^{-0.03}.
\end{aligned}$$

◀

Proof of Claim 17. For each $x \in A$ consider the indicator random variable $\mathbf{1}_x \in \{0,1\}$ indicating whether $x \in \boldsymbol{U}$. Let $\boldsymbol{s} \coloneqq \sum_{x \in A} \mathbf{1}_x$ so that $\boldsymbol{s} = |A \cap \boldsymbol{U}|$ and $\mathbf{E}[\boldsymbol{s}] = \delta|A|$, where $\delta = |\boldsymbol{U}|/m^n = m^{-0.64n}$. We use the second moment method to estimate

$$\mathbf{Pr}[A \cap \boldsymbol{U} \neq \emptyset] \;=\; 1 - \mathbf{Pr}[\boldsymbol{s} = 0] \;\geq\; 1 - \frac{\mathbf{Var}[\boldsymbol{s}]}{\mathbf{E}[\boldsymbol{s}]^2}.$$

Thus, to prove the claim it suffices to show that $\mathbf{Var}[\boldsymbol{s}] \leq 2m^{-0.01} \cdot \mathbf{E}[\boldsymbol{s}]^2 = 2m^{-0.01} \cdot \delta^2|A|^2$. Since

$$\mathbf{Var}[\boldsymbol{s}] \;=\; \textstyle\sum_{x,x'} \mathbf{Cov}[\mathbf{1}_x, \mathbf{1}_{x'}] \;=\; \sum_{x,x'} \left(\mathbf{E}[\mathbf{1}_x \mathbf{1}_{x'}] - \mathbf{E}[\mathbf{1}_x]\mathbf{E}[\mathbf{1}_{x'}]\right),$$

it suffices to show that, for each fixed $x^* \in A$,

$$\textstyle\sum_{x \in A} \mathbf{Cov}[\mathbf{1}_x, \mathbf{1}_{x^*}] \;\leq\; 2m^{-0.01} \cdot \delta^2|A|.$$

Fix $x^* \in A$. Let $I_x \subseteq [n]$ denote the set of all blocks i such that $x_i = x^*_i$. First note that under $I_x = \emptyset$ it holds that $\mathbf{Cov}[\mathbf{1}_x, \mathbf{1}_{x^*}] < 0$, i.e., the events "$x \in \boldsymbol{U}$" and "$x^* \in \boldsymbol{U}$" are negatively correlated. The interesting case is thus $I_x \neq \emptyset$ when the two events are positively correlated. We note that

$$\mathbf{Pr}[x \in \boldsymbol{U} \mid x^* \in \boldsymbol{U}] \;=\; \left(\tfrac{m^{0.36}-1}{m-1}\right)^{n-|I_x|} \;\leq\; m^{0.64|I_x|} \cdot \delta. \tag{2}$$

Let $\boldsymbol{I}$ be the distribution of $I_{\boldsymbol{x}}$ when $\boldsymbol{x} \in A$ is chosen uniformly at random. We have

$$\begin{aligned}
\textstyle\sum_{x \in A} \mathbf{Cov}[\mathbf{1}_x, \mathbf{1}_{x^*}] &\leq \textstyle\sum_{x : I_x \neq \emptyset} \mathbf{Cov}[\mathbf{1}_x, \mathbf{1}_{x^*}] \\
&\leq \textstyle\sum_{x : I_x \neq \emptyset} \mathbf{E}[\mathbf{1}_x \mathbf{1}_{x^*}] \\
&= \textstyle\sum_{x : I_x \neq \emptyset} \mathbf{Pr}[x \in \boldsymbol{U} \text{ and } x^* \in \boldsymbol{U}] \\
&= \mathbf{Pr}[x^* \in \boldsymbol{U}] \cdot \textstyle\sum_{x : I_x \neq \emptyset} \mathbf{Pr}[x \in \boldsymbol{U} \mid x^* \in \boldsymbol{U}] \\
&= \delta \cdot \textstyle\sum_{x : I_x \neq \emptyset} \mathbf{Pr}[x \in \boldsymbol{U} \mid x^* \in \boldsymbol{U}] \\
&= \delta|A| \cdot \textstyle\sum_{\emptyset \neq I \subseteq [n]} \mathbf{Pr}[\boldsymbol{I} = I] \cdot \mathbf{E}_{\boldsymbol{x} \sim A | I_{\boldsymbol{x}} = I} \mathbf{Pr}[\boldsymbol{x} \in \boldsymbol{U} \mid x^* \in \boldsymbol{U}] \\
&\leq \delta|A| \cdot \textstyle\sum_{\emptyset \neq I \subseteq [n]} \mathbf{Pr}_{\boldsymbol{x} \sim A}[\boldsymbol{x}_I = x^*_I] \cdot \mathbf{E}_{\boldsymbol{x} \sim A | I_{\boldsymbol{x}} = I} \mathbf{Pr}[\boldsymbol{x} \in \boldsymbol{U} \mid x^* \in \boldsymbol{U}] \\
&\leq \delta|A| \cdot \textstyle\sum_{\emptyset \neq I \subseteq [n]} 2^{-0.9|I| \log m} \cdot m^{0.64|I|} \cdot \delta \qquad \text{(0.9-density and (2))} \\
&= \delta^2|A| \cdot \textstyle\sum_{\emptyset \neq I \subseteq [n]} 2^{-0.26|I| \log m} \\
&= \delta^2|A| \cdot \textstyle\sum_{k \in [n]} \binom{n}{k} 2^{-0.26k \log m} \\
&\leq \delta^2|A| \cdot \textstyle\sum_{k \in [n]} (m^{0.25})^k \cdot 2^{-0.26k \log m} \\
&\leq \delta^2|A| \cdot 2 \cdot 2^{-0.01 \log m} \\
&\leq 2m^{-0.01} \cdot \delta^2|A|.
\end{aligned}$$

◀

4 Application

In this section we prove Theorem 3, restated here for convenience.

▶ **Theorem 3.** *There exists a total* $F\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ *satisfying the following.*

- *F has large P^{NP} communication complexity:* $\mathsf{P}^{\mathsf{NPcc}}(F) \geq n^{\Omega(1)}$.
- *For any product distribution μ over $\{0,1\}^n \times \{0,1\}^n$, there exists a monochromatic rectangle R that is large:* $\log(1/\mu(R)) \leq \log^{O(1)} n$.

The function witnessing the separation is $F \coloneqq f \circ g^n$ where $g \coloneqq \textsc{Ind}_m$ is the index function with $m \coloneqq n^4$ and $f\colon \{0,1\}^n \to \{0,1\}$ is defined as follows. We interpret the input M to f as a $\sqrt{n} \times \sqrt{n}$ boolean matrix, and set

$$f(M) \;\coloneqq\; 1 \qquad \text{iff} \qquad \text{every row of } M \text{ contains a unique 1-entry.}$$

Complexity class aficionados [1] can recognize f as the canonical complete problem for the decision tree analogue of $\forall \cdot \mathsf{US}$ ($\subseteq \Pi_2\mathsf{P}$) where US is the class of functions whose 1-inputs admit a *unique* witness [5]. We have $F\colon \{0,1\}^{n \log m} \times \{0,1\}^{nm} \to \{0,1\}$, but we can polynomially pad Alice's input length to match Bob's (as in the statement of Theorem 3).

4.1 Lower bound

It is proved in several sources [32, 21, 16] that f cannot be computed by an efficient $\Sigma_2\mathsf{P}$-type decision tree (i.e., quasi-polynomial-size depth-3 circuit with an Or-gate at the top and small bottom fan-in), let alone an efficient P^{NP} decision tree. However, for completeness, we might as well give a simple proof using our characterization (Theorem 11). Applying the lifting theorem to the following lemma yields the lower bound.

▶ **Lemma 19.** $\mathsf{DL}^{\mathsf{dt}}(f) \geq \sqrt{n}$.

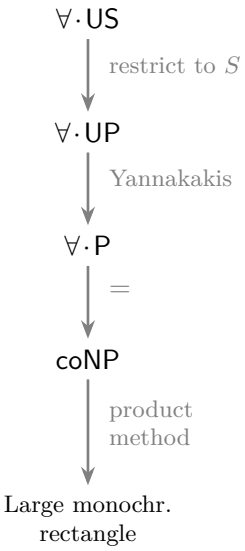
Proof. By Theorem 11 it is enough to exhibit a nonempty subset $Z \subseteq \{0,1\}^n$ of inputs such that each conjunction C of width $\sqrt{n} - 1$ accepts an input in $Z_1 \coloneqq Z \cap f^{-1}(1)$ iff it accepts an input in $Z_0 \coloneqq Z \cap f^{-1}(0)$. We define Z as the set of $\sqrt{n} \times \sqrt{n}$ matrices with at most one 1-entry in each row. If C accepts an input $M \in Z_1$, then there is some row of M none of whose entries are read by C; we may modify that row to all-0 and conclude that C accepts an input in Z_0. If C accepts an input $M \in Z_0$, then for each all-0 row of M there is some entry that is not read by C; we may modify each of those entries to a 1 and conclude that C accepts an input in Z_1. ◀

4.2 Upper bound

Let μ be a product distribution over the domain of $F = f \circ g^n$. Call a matrix M *heavy* if it contains a row with at least two 1-entries. Hence $f(M) = 0$ for every heavy matrix M. There is an efficient nondeterministic protocol of cost $k \leq O(\log n)$, call it Π, that checks whether a particular (x, y) describes a heavy matrix $M = g^n(x, y)$. Namely, Π guesses a row index $i \in [\sqrt{n}]$ and two column indices $1 \leq j < j' \leq \sqrt{n}$, and then communicates $2 \log m + 1 \leq O(\log n)$ bits to check that $M_{ij} = M_{ij'} = 1$. We view Π as defining a rectangle covering $\bigcup_{i \in [2^k]} R_i$ of all those (x, y) that describe heavy matrices. Note that each R_i is monochromatic for F.

If there is an R_i with $\mu(R_i) \geq 2^{-4k}$, the theorem is proved. So suppose not: $\mu(R_i) < 2^{-4k}$ for all i. Starting with $S \coloneqq$ domain of F and iterating over the R_i exactly as in the proof of Theorem 10, we can delete from S either the rows or the columns of each R_i, ending up with

a rectangle S still of measure $\mu(S) \geq 1 - 2^k \cdot 2^{-2k} \geq 0.99$. We will complete the argument by showing that F_S (i.e., F restricted to the rectangle S) admits a large monochromatic rectangle relative to μ_S, the conditional distribution of μ given S (which is also product).



All $(x, y) \in S$ are such that $M = g^n(x, y)$ is *not* heavy. This means that the function F_S is easier than the ($\forall \cdot \mathsf{US}$-complete) function F in the following sense: for each row $i \in [\sqrt{n}]$ there is an efficient $O(\log n)$-cost nondeterministic protocol, call it Π_i, to check whether the i-th row of $M = g^n(x, y)$ contains a 1-entry, and moreover, this protocol is *unambiguous* in that it has at most one accepting computation on any input. (In complexity lingo, F_S admits an efficient $\forall \cdot \mathsf{UP}$ protocol.) It is a well-known theorem of Yannakakis [40, Lemma 1] that any such unambiguous Π_i can be made deterministic with at most a quadratic blow-up in cost; let Π_i^{det} be that $O(\log^2 n)$-bit deterministic protocol. But now $\neg F_S$ (negation of F_S) is computed by the following $O(\log^2 n)$-bit nondeterministic protocol: on input (x, y) guess a row index $i \in [\sqrt{n}]$ and run Π_i^{det} accepting iff $\Pi_i^{\text{det}}(x, y) = 0$. (That is, F_S admits an efficient $\forall \cdot \mathsf{P} = \mathsf{coNP}$ protocol.) We proved $\mathsf{NP}^{\mathsf{cc}}(\neg F_S) \leq O(\log^2 n)$; in particular,

$$\mathsf{DL}^{\mathsf{cc}}(F_S) \;\leq\; O(\mathsf{P}^{\mathsf{NPcc}}(F_S)) \;\leq\; O(\mathsf{NP}^{\mathsf{cc}}(\neg F_S)) \;\leq\; O(\log^2 n).$$

Hence we can apply (as a black box) the product method (Theorem 10) to find a monochromatic rectangle $R \subseteq S$ with $\log(1/\mu_S(R)) \leq O(\log^2 n)$ and hence $\log(1/\mu(R)) \leq O(\log^2 n)$. This completes the proof of Theorem 3.

5 Conclusion

Let $\mathsf{PM}(F)$ denote the best lower bound on $\mathsf{DL}^{\mathsf{cc}}(F)$ that can be derived by the product method (Theorem 10). For any communication complexity measure $\mathcal{C}(F)$, we use the convention that $\mathcal{C}$ by itself refers to the class of (families of) functions $F\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ with $\mathcal{C}(F) \leq \text{polylog}(n)$. Then our application (Theorem 3) shows that the inclusion $\mathsf{P}^{\mathsf{NPcc}} \subseteq \mathsf{PM}$ is strict: there is an $F \in \mathsf{PM} \setminus \mathsf{P}^{\mathsf{NPcc}}$. Here are some open questions.

1. Is there an $F \in \mathsf{PM} \setminus \mathsf{UPP}^{\mathsf{cc}}$? This would be a stronger result since $\mathsf{P}^{\mathsf{NPcc}} \subseteq \mathsf{UPP}^{\mathsf{cc}}$. Note that our $\forall \cdot \mathsf{US}$-complete function does not witness this, since it is in $\mathsf{PP}^{\mathsf{cc}}$. One way to see this is to note that it is the intersection of a $\mathsf{coNP}^{\mathsf{cc}}$ function (does each row have at most one 1?) and a $\mathsf{PP}^{\mathsf{cc}}$ function (is the number of 1's at least the number of rows?), and use the closure of PP under intersection [4].

2. Is there any reasonable upper bound for PM? For example, does $\mathsf{PM} \subseteq \mathsf{PSPACE}^{\mathsf{cc}}$ hold?
3. Does $\mathsf{BPP}^{\mathsf{cc}} \subseteq \mathsf{PM}$ or even $\mathsf{BPP}^{\mathsf{cc}} \subseteq \mathsf{P}^{\mathsf{NPcc}}$ hold for total functions? The separation $\mathsf{BPP}^{\mathsf{cc}} \not\subseteq \mathsf{PM}$ was shown for partial functions implicitly in [25].
4. Is there a lossless $\mathsf{P}^{\mathsf{NPdt}}$-to-$\mathsf{P}^{\mathsf{NPcc}}$ lifting theorem (Conjecture 7)?
5. Can the quadratic upper bounds in Facts 4 and 5 be shown tight for more general parameters (beyond constant $\mathsf{DL}^{\mathsf{dt}}(f)$ and logarithmic $\mathsf{DL}^{\mathsf{cc}}(F)$ as in subsection 1.4)?

Acknowledgments. We thank Paul Balister, Shalev Ben-David, Béla Bollobás, Robin Kothari, Nirman Kumar, Santosh Kumar, Govind Ramnarayan, Madhu Sudan, Li-Yang Tan, and Justin Thaler for discussions and correspondence.

References

1 Scott Aaronson, Greg Kuperberg, and Christopher Granade. Complexity zoo. Online, 2017. URL: https://complexityzoo.uwaterloo.ca.

2 László Babai, Peter Frankl, and Janos Simon. Complexity classes in communication complexity theory. In *Proceedings of the 27th Symposium on Foundations of Computer Science (FOCS)*, pages 337–347. IEEE, 1986. doi:10.1109/SFCS.1986.15.

3 Richard Beigel. Perceptrons, PP, and the polynomial hierarchy. *Computational complexity*, 4(4):339–349, 1994. doi:10.1007/BF01263422.

4 Richard Beigel, Nick Reingold, and Daniel Spielman. PP is closed under intersection. *Journal of Computer and System Sciences*, 50(2):191–202, 1995. doi:10.1006/jcss.1995.1017.

5 Andreas Blass and Yuri Gurevich. On the unique satisfiability problem. *Information and Control*, 55(1–3):80–88, 1982. doi:10.1016/S0019-9958(82)90439-9.

6 Harry Buhrman and Ronald de Wolf. Complexity measures and decision tree complexity: A survey. *Theoretical Computer Science*, 288(1):21–43, 2002. doi:10.1016/S0304-3975(01)00144-X.

7 Harry Buhrman, Nikolai Vereshchagin, and Ronald de Wolf. On computation and communication with small bias. In *Proceedings of the 22nd Conference on Computational Complexity (CCC)*, pages 24–32. IEEE, 2007. doi:10.1109/CCC.2007.18.

8 Mark Bun and Justin Thaler. Approximate degree and the complexity of depth three circuits. Technical Report TR16-121, Electronic Colloquium on Computational Complexity (ECCC), 2016. URL: https://eccc.weizmann.ac.il/report/2016/121/.

9 Siu On Chan, James Lee, Prasad Raghavendra, and David Steurer. Approximate constraint satisfaction requires large LP relaxations. *Journal of the ACM*, 63(4):34:1–34:22, 2016. doi:10.1145/2811255.

10 Susanna 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 Symposium on Foundations of Computer Science (FOCS)*, pages 295–304. IEEE, 2016. doi:10.1109/FOCS.2016.40.

11 Mika Göös. Lower bounds for clique vs. independent set. In *Proceedings of the 56th Symposium on Foundations of Computer Science (FOCS)*, pages 1066–1076. IEEE, 2015. doi:10.1109/FOCS.2015.69.

12 Mika Göös, Pritish Kamath, Toniann Pitassi, and Thomas Watson. Query-to-Communication Lifting for P^{NP}. *Electronic Colloquium on Computational Complexity (ECCC)*, 24:24, 2017. URL: https://eccc.weizmann.ac.il/report/2017/024.

13 Mika Göös, Shachar Lovett, Raghu Meka, Thomas Watson, and David Zuckerman. Rectangles are nonnegative juntas. *SIAM Journal on Computing*, 45(5):1835–1869, 2016. doi:10.1137/15M103145X.

14 Mika Göös, Toniann Pitassi, and Thomas Watson. Deterministic communication vs. partition number. In *Proceedings of the 56th Symposium on Foundations of Computer Science (FOCS)*, pages 1077–1088. IEEE, 2015. doi:10.1109/FOCS.2015.70.

15 Mika Göös, Toniann Pitassi, and Thomas Watson. The landscape of communication complexity classes. In *Proceedings of the 43rd International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 55 of *LIPIcs*, pages 86:1–86:15. Schloss Dagstuhl, 2016. doi:10.4230/LIPIcs.ICALP.2016.86.

16 Johan Håstad, Stasys Jukna, and Pavel Pudlák. Top-down lower bounds for depth-three circuits. *Computational Complexity*, 5(2):99–112, 1995. doi:10.1007/BF01268140.

17 Hamed Hatami, Kaave Hosseini, and Shachar Lovett. Structure of protocols for XOR functions. In *Proceedings of the 57th Symposium on Foundations of Computer Science (FOCS)*, pages 282–288. IEEE, 2016. doi:10.1109/FOCS.2016.38.

18 Russell Impagliazzo and Ryan Williams. Communication complexity with synchronized clocks. In *Proceedings of the 25th Conference on Computational Complexity (CCC)*, pages 259–269. IEEE, 2010. doi:10.1109/CCC.2010.32.

19 Stasys Jukna. *Boolean Function Complexity: Advances and Frontiers*, volume 27 of *Algorithms and Combinatorics*. Springer, 2012.

20 Mauricio Karchmer, Eyal Kushilevitz, and Noam Nisan. Fractional covers and communication complexity. *SIAM Journal on Discrete Mathematics*, 8(1):76–92, 1995. doi:10.1137/S0895480192238482.

21 Ker-I Ko. Separating and collapsing results on the relativized probabilistic polynomial-time hierarchy. *Journal of the ACM*, 37(2):415–438, 1990. doi:10.1145/77600.77623.

22 Pravesh 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 Symposium on Theory of Computing (STOC)*. ACM, 2017. To appear.

23 Eyal Kushilevitz and Noam Nisan. *Communication Complexity*. Cambridge University Press, 1997.

24 James Lee, Prasad Raghavendra, and David Steurer. Lower bounds on the size of semidefinite programming relaxations. In *Proceedings of the 47th Symposium on Theory of Computing (STOC)*, pages 567–576. ACM, 2015. doi:10.1145/2746539.2746599.

25 Periklis Papakonstantinou, Dominik Scheder, and Hao Song. Overlays and limited memory communication. In *Proceedings of the 29th Conference on Computational Complexity (CCC)*, pages 298–308. IEEE, 2014. doi:10.1109/CCC.2014.37.

26 Ramamohan Paturi and Janos Simon. Probabilistic communication complexity. *Journal of Computer and System Sciences*, 33(1):106–123, 1986. doi:10.1016/0022-0000(86)90046-2.

27 Anup Rao and Amir Yehudayoff. *Communication Complexity*. In preparation, 2017.

28 Ran Raz and Pierre McKenzie. Separation of the monotone NC hierarchy. *Combinatorica*, 19(3):403–435, 1999. doi:10.1007/s004930050062.

29 Alexander Razborov and Alexander Sherstov. The sign-rank of AC^0. *SIAM Journal on Computing*, 39(5):1833–1855, 2010. doi:10.1137/080744037.

30 Ronald Rivest. Learning decision lists. *Machine Learning*, 2(3):229–246, 1987. doi:10.1007/BF00058680.

31 Robert Robere, Toniann Pitassi, Benjamin Rossman, and Stephen Cook. Exponential lower bounds for monotone span programs. In *Proceedings of the 57th Symposium on Foundations of Computer Science (FOCS)*, pages 406–415. IEEE, 2016. doi:10.1109/FOCS.2016.51.

32 Miklos Santha. Relativized Arthur–Merlin versus Merlin–Arthur games. *Information and Computation*, 80(1):44–49, 1989. doi:10.1016/0890-5401(89)90022-9.

33 Rocco Servedio, Li-Yang Tan, and Justin Thaler. Attribute-efficient learning and weight-degree tradeoffs for polynomial threshold functions. In *Proceedings of the 25th Conference*

on Learning Theory (COLT), pages 14.1–14.19. JMLR, 2012. URL: http://www.jmlr.org/proceedings/papers/v23/servedio12/servedio12.pdf.

34 Alexander Sherstov. The pattern matrix method. *SIAM Journal on Computing*, 40(6):1969–2000, 2011. doi:10.1137/080733644.

35 Yaoyun Shi and Yufan Zhu. Quantum communication complexity of block-composed functions. *Quantum Information and Computation*, 9(5–6):444–460, 2009.

36 Justin Thaler. Lower bounds for the approximate degree of block-composed functions. In *Proceedings of the 43rd International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 55 of *LIPIcs*, pages 17:1–17:15. Schloss Dagstuhl, 2016. doi:10.4230/LIPIcs.ICALP.2016.17.

37 Salil Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 7(1–3):1–336, 2012. doi:10.1561/0400000010.

38 Nikolai Vereshchagin. Relativizability in complexity theory. In *Provability, Complexity, Grammars*, volume 192 of *AMS Translations, Series 2*, pages 87–172. American Mathematical Society, 1999.

39 Ryan Williams. Brute force search and oracle-based computation. Technical report, Cornell University, 2001. URL: https://web.stanford.edu/~rrwill/bfsearch-rev.ps.

40 Mihalis Yannakakis. Expressing combinatorial optimization problems by linear programs. *Journal of Computer and System Sciences*, 43(3):441–466, 1991. doi:10.1016/0022-0000(91)90024-Y.

Improved Bounds for Quantified Derandomization of Constant-Depth Circuits and Polynomials*†

Roei Tell

Department of Computer Science and Applied Mathematics, Weizmann Institute of Science, Rehovot, Israel
roei.tell@weizmann.ac.il

Abstract

This work studies the question of *quantified derandomization*, which was introduced by Goldreich and Wigderson (STOC 2014). The generic quantified derandomization problem is the following: For a circuit class $\mathcal{C}$ and a parameter $B = B(n)$, given a circuit $C \in \mathcal{C}$ with n input bits, decide whether C rejects all of its inputs, or accepts all but $B(n)$ of its inputs. In the current work we consider three settings for this question. In each setting, we bring closer the parameter setting for which we can unconditionally construct relatively fast quantified derandomization algorithms, and the "threshold" values (for the parameters) for which any quantified derandomization algorithm implies a similar algorithm for standard derandomization.

For **constant-depth circuits**, we construct an algorithm for quantified derandomization that works for a parameter $B(n)$ that is only *slightly smaller* than a "threshold" parameter, and is significantly faster than the best currently-known algorithms for standard derandomization. On the way to this result we establish a new derandomization of the switching lemma, which significantly improves on previous results when the width of the formula is small. For **constant-depth circuits with parity gates**, we lower a "threshold" of Goldreich and Wigderson from depth five to depth four, and construct algorithms for quantified derandomization of a remaining type of layered depth-3 circuit that they left as an open problem. We also consider the question of constructing hitting-set generators for multivariate **polynomials over large fields that vanish rarely**, and prove two lower bounds on the seed length of such generators.

Several of our proofs rely on an interesting technique, which we call the *randomized tests* technique. Intuitively, a standard technique to deterministically find a "good" object is to construct a simple deterministic test that decides the set of good objects, and then "fool" that test using a pseudorandom generator. We show that a similar approach works also if the simple deterministic test is replaced with a *distribution over simple tests*, and demonstrate the benefits in using a distribution instead of a single test.

1998 ACM Subject Classification F.1.2 Models of Computation, F.1.3 Complexity Measures and Classes

Keywords and phrases Computational complexity, derandomization, quantified derandomization, hitting-set generator, constant-depth circuits

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.13

* An online version of this paper is available at ECCC [21], https://eccc.weizmann.ac.il/report/2016/191/.

† This research was partially supported by the Minerva Foundation with funds from the Federal German Ministry for Education and Research. The research was also partially supported by Irit Dinur's ERC grant number 239986.

32nd Computational Complexity Conference (CCC 2017).
Editor: Ryan O'Donnell; Article No. 13; pp. 13:1–13:48

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

1 Introduction

For a circuit class $\mathcal{C}$, the standard (one-sided error) derandomization problem is the following: Given a circuit $C \in \mathcal{C}$, distinguish in deterministic polynomial time between the case that C rejects all of its inputs and the case that C accepts most of its inputs. Impagliazzo and Wigderson [12], following Nisan and Wigderson [15], showed that under reasonable complexity-theoretic assumptions, the standard derandomization problem can be solved even for a class as large as $\mathcal{C} = \mathcal{P}/\text{poly}$. However, at this time we do not know how to unconditionally solve this problem even when $\mathcal{C}$ is the class of polynomial-sized CNFs.

A couple of years ago, Goldreich and Wigderson [9] put forward a potentially easier problem, which they call *quantified* derandomization. Given a class $\mathcal{C}$ and a parameter $B = B(n)$, the problem is to decide whether a circuit $C \in \mathcal{C}$ over n input bits rejects all of its inputs, or accepts *all but* $B(n)$ *of its inputs* (rather than just "most" of its inputs). We call $B(n)$ the "badness" parameter, since it represents the number of bad random strings (i.e., the ones that lead the algorithm to an incorrect decision). Indeed, the standard derandomization problem is captured by the parameter $B(n) = 2^n/2$, but we are typically interested in $B(n)$'s that are much smaller. On the other hand, polynomially-bounded values (e.g., $B(n) = O(n)$) can be easily handled by an algorithm that simply evaluates C on $B(n) + 1$ fixed inputs.

Goldreich and Wigderson constructed algorithms that solve the quantified derandomization problem for various classes $\mathcal{C}$ and parameters $B = B(n)$. For example, they constructed a polynomial time hitting-set generator for $\mathcal{AC}^0$ circuits that accept all but $B(n) = 2^{n^{1-\epsilon}}$ of their inputs, for any $\epsilon > 0$. On the other hand, they showed that for some classes $\mathcal{C}$ and a sufficiently high badness parameter $B(n)$, the quantified derandomization problem is as difficult as the standard derandomization problem (since the latter can be reduced to the former). We call such parameter values threshold values, since a quantified derandomization with a badness parameter $B(n)$ that surpasses this threshold will yield a result for a standard derandomization problem.

Our contributions in this work are of two types. On the one hand, we construct quantified derandomization algorithms that work for a broader range of parameters, compared to [9] (e.g., larger values of $B(n)$, or broader circuit classes). On the other hand, we show that quantified derandomization of circuit classes that are more limited (compared to [9]) is still at least as difficult as certain standard derandomization problems.

The "take-home" message: Considered together, our results bring closer two settings of parameters: The parameter setting for which we can unconditionally construct relatively fast quantified derandomization algorithms, and the "threshold" values (for the parameters) for which any quantified derandomization algorithm implies a similar algorithm for standard derandomization.

1.1 Brief overview of our results

Let us informally state the main results in this work, which we later outline in more detail:

- **Constant-depth circuits (see Section 1.2):** For circuits of depth D, the badness parameter $B(n) = \exp\left(n/\log^{D-O(1)}(n)\right)$ is a threshold value, since an algorithm for quantified derandomization with such a $B(n)$ implies an algorithm for *standard* derandomization of circuits of smaller depth $d \leq D - 12$ (see Theorem 1).
  We show that taking $B(n)$ to be only *slightly smaller* than the threshold value allows for derandomization that is significantly faster than the best currently-known standard

derandomization. Specifically, we construct a hitting-set generator for depth-D circuits with badness $B(n) = \exp\left(n/\log^{D-2}(n)\right)$ that has seed length $\tilde{O}(\log^3(n))$; in particular, the seed length *does not depend on the depth* D (see Theorem 2).
The latter is a special case of a more general result that we prove, which extends the main theorem of Goldreich and Wigderson [9]: We establish a trade-off between the badness parameter and the seed length of hitting-set generators for $\mathcal{AC}^0$. This is done by constructing a parametrized hitting-set generator that can work with large badness parameters, at the expense of a super-logarithmic seed (see Theorem 3). The key part in this construction is a *new derandomization of the switching lemma*, which is our main technical contribution in the context of constant-depth circuits. The seed length in the new derandomization is significantly shorter than in previous derandomizations when the width w of the formula is small (i.e., $w = o(\log(n))$).

- **Constant-depth circuits with parity gates (see Section 1.3):** We show that a threshold for derandomization of $\mathcal{AC}^0[\oplus]$ exists at depth four with the parameter 2^{n^c}, for any $c > 0$. Hence, an appealing frontier is $\mathcal{AC}^0[\oplus]$ circuits of depth three with the parameter $B(n) = 2^{n^c}$. Goldreich and Wigderson derandomized various types of such circuits, and left one last type as an open problem. We make significant progress on the last remaining type: Specifically, we construct a whitebox hitter for circuits with a top $\oplus$ gate, a middle layer of $\wedge$ gates, and a bottom layer of $\oplus$ gates, under various sub-quadratic bounds on the number of gates in the different layers (see Theorem 6).
  We also affirm a conjecture from [9], by showing a reduction of the problem of hitting such $\oplus \wedge \oplus$-circuits to the problem of hitting biased $\mathbb{F}_2$-polynomials of *bounded (non-constant) degree* (see Theorem 7).
- **Polynomials that vanish rarely (see Section 1.4):** We study the problem of constructing hitting-set generators for polynomials $\mathbb{F}^n \to \mathbb{F}$ that *vanish rarely*, where $\mathbb{F}$ is an arbitrary finite field. We prove two lower bounds on the seed length of such hitting-set generators. The main result is that any hitting-set generator for degree-d polynomials that vanish on at most $1/\text{poly}(|\mathbb{F}|)$ of their inputs requires a seed of length similar to that of hitting-set generators for *all* degree-d polynomials (see Theorem 8).
  As part the proofs, we reduce the task of constructing a hitting-set generator for degree-d polynomials to the task of constructing a hitting-set generator for polynomials of degree d' that vanish rarely, where $d \leq d' \leq \text{poly}(d)$; this is a form of "error reduction" for polynomials that incurs only a mild increase in the degree.

Several of our results are based on a general technique that might be of independent interest, which we call the **randomized tests** technique (see Section 2.1). Intuitively, a standard approach to deterministically find an object in some predetermined set $G \subseteq \{0,1\}^n$ is to construct a simple deterministic test that decides G, and then "fool" the test using a pseudorandom generator. We show that a similar approach works if the simple deterministic test is replaced with a *distribution over simple tests*, and the pseudorandom generator is required to "fool" the residual deterministic tests. In many settings, the fact that we use randomness (i.e., use a distribution over tests) yields residual tests that are simpler than any corresponding deterministic test (see Section 2.2 for a concrete example).

Towards stating the results, recall that a hitting-set generator for a class of functions $\mathcal{F}$ from $\{0,1\}^n$ to $\{0,1\}$ is an algorithm $G : \{0,1\}^\ell \to \{0,1\}^n$, for some $\ell = \ell(n)$, such that for every $f \in \mathcal{F}$ there exists some $s \in \{0,1\}^\ell$ such that $f(G(s)) \neq 0$. We say that the hitting-set generator has density $\epsilon > 0$ if for every $f \in \mathcal{F}$ it holds that $\Pr_{s\in\{0,1\}^\ell}[f(G(s)) \neq 0] \geq \epsilon$ (see Definition 10). The definition of hitting-set generators extends naturally to functions $\mathbb{F}^n \to \mathbb{F}$, for any field $\mathbb{F}$ (see Definition 11).

1.2 Constant-depth circuits

Let us first state the threshold values for quantified derandomization of $\mathcal{AC}^0$, and then turn to describe our algorithms for quantified derandomization. Goldreich and Wigderson showed that the value $B(n) = 2^{n/\log^{0.99 \cdot D}(n)}$ is a threshold value for quantified derandomization of depth-D circuits. Specifically, they reduced the *standard* derandomization problem of depth-d circuits to the problem of quantified derandomization of circuits of depth $D \gg d$ with $B(n) = 2^{n/\log^{D-O(d)}(n)}$ (see [9, Thm 3.4 (full version)]). Since their work, Cheng and Li [5] improved the known techniques for error-reduction within $\mathcal{AC}^0$, which allows us to further decrease the threshold value, as follows:

▶ **Theorem 1** (Threshold for Quantified Derandomization of $\mathcal{AC}^0$)**.** *For any $d \geq 2$ and $D > d+11$, the standard derandomization problem of depth-d circuits reduces in deterministic polynomial-time to the quantified derandomization problem of circuits of depth D that accept all but $B(n) = 2^{n/\log^{D-d-11}(n)}$ of their inputs.*

Our main result for $\mathcal{AC}^0$ circuits is a derandomization of depth-D circuits with the badness parameter $B(n) = 2^{n/\log^{D-2}(n)}$, which *is only slightly smaller* than the threshold value in Theorem 1. The quantified derandomization algorithm runs in time that is significantly faster than the current state-of-the-art for derandomizing $\mathcal{AC}^0$:

▶ **Theorem 2** (Quantified Derandomization of $\mathcal{AC}^0$ with Badness $2^{n/\log^{D-2}(n)}$)**.** *For any $D \geq 2$, there exists a hitting-set generator with seed length $\tilde{O}(\log^3(n))$ for the class of depth-D circuits over n input bits that accept all but at most $B(n) = 2^{\Omega\left(n/\log^{D-2}(n)\right)}$ of their inputs.*

We stress that the power of the poly-logarithm in the seed length in Theorem 2 does not depend on the depth D. Any *standard* hitting-set generator for $\mathcal{AC}^0$ (i.e., with $B(n) = 2^n/2$) with such a seed length would be a major breakthrough, and in particular would significantly improve the lower bounds of Håstad for $\mathcal{AC}^0$ [11] (see, e.g., [24, Prob. 7.1] and [23, "Barriers to Further Progress"]).

The badness parameters in Theorems 1 and 2 are indeed very close, yet the smaller badness parameter allows for derandomization in time $2^{\tilde{O}(\log^3(n))}$ whereas the larger badness parameter is a threshold for standard derandomization. This represents a progress towards the goal of the quantified derandomization approach, which is to *close* the gap between the two parameters: That is, to either increase the badness parameter in Theorem 2, or decrease the parameter in Theorem 1, and obtain a standard derandomization of $\mathcal{AC}^0$.

Theorem 2 is a special case of the following, more general result, which extends the main theorem of Goldreich and Wigderson [9]. Their algorithm works with logarithmic seed and badness parameter $B(n) = 2^{n^{1-\Omega(1)}}$. The following result is parametrized (by the parameter t), and can work with badness parameters that are larger than $2^{n^{1-\Omega(1)}}$, at the expense of a longer (i.e., super-logarithmic) seed; Theorem 2 is the special case where both the badness parameter and the seed are the largest possible in this result.

▶ **Theorem 3** (Quantified Derandomization of $\mathcal{AC}^0$: A General Trade-Off)**.** *For any $D \geq 2$ and $t : \mathbb{N} \to \mathbb{N}$ such that $t(n) \leq O(\log(n))$, there exists a hitting-set generator that uses a seed of length $\tilde{O}(t^2 \cdot \log(n))$ for the class of depth-D circuits over n input bits that accept all but at most $B(n) = \exp\left(n^{1-1/\Omega(t)}/t^{d-2}\right)$ of their inputs.*

Indeed, the main result in [9] is essentially obtained (up to a $\operatorname{poly}\log\log(n)$ factor in the seed length) by setting $t = O(1)$, whereas Theorem 2 is obtained by setting $t = O(\log(n))$. Theorem 3 is based on a *new derandomization of Hastad's switching lemma*, which is our main technical contribution in this section.

▶ **Proposition 4** (New Derandomization of the Switching Lemma; Informal). *Let $n \in \mathbb{N}$ and $w \leq O(\log(n))$. Then, there exists an algorithm that on an input random seed of length $\tilde{O}(w^2 \cdot \log(n))$ outputs a restriction $\rho \in \{0,1,\star\}^n$ such that for every depth-2 formula $F : \{0,1\}^n \to \{0,1\}$ of size $\mathrm{poly}(n)$ and width w the following holds:*

- *There exist two formulas $F^{\mathtt{low}}$ and $F^{\mathtt{up}}$ such that for every $x \in \{0,1\}^n$ it holds that $F^{\mathtt{low}}(x) \leq F(x) \leq F^{\mathtt{up}}(x)$.*
- *With probability $1 - 1/\mathrm{poly}(n)$ it holds that both $F^{\mathtt{low}}{\restriction_\rho}$ and $F^{\mathtt{up}}{\restriction_\rho}$ can be computed by decision trees of depth $O(\log(n))$, and that both $F^{\mathtt{low}}{\restriction_\rho}$ and $F^{\mathtt{up}}{\restriction_\rho}$ agree with F on $1 - 1/\mathrm{poly}(n)$ of the inputs in the subcube that corresponds to the living variables under ρ.*

Note that the seed length of the algorithm from Proposition 4 depends on the width of the formula F. Previous derandomizations of the switching lemma can also be adapted to depend on the width, but when the width is $o(\log(n))$ the seed length in Proposition 4 is significantly shorter than in these adaptations; see Section 2.2 for further details.

1.3 Constant-depth circuits with parity gates

The next circuit class that we study is that of constant-depth circuits that also have gates computing the parity function or the negated parity function (i.e., $\mathcal{AC}^0[\oplus]$). Specifically, we consider $\mathcal{AC}^0[\oplus]$ circuits that are *layered*, in the sense that all gates at a particular distance from the input gates are of the same gate-type.

We first observe that the standard derandomization problem of CNFs can be reduced to the problem of derandomizing layered $\mathcal{AC}^0[\oplus]$ circuits of *depth four* with $B(n) = 2^{n^c}$, which yields a "threshold" at depth four with such a badness parameter. This improves on a similar result of [9] that refers to depth five.

▶ **Theorem 5** (A Threshold for Quantified Derandomization of $\mathcal{AC}^0[\oplus]$ at Depth Four). *Assume that, for some $c > 0$, there exists a polynomial-time algorithm A such that, when A is given as input a layered depth-four $\mathcal{AC}^0[\oplus]$ circuit C over n input bits that accepts all but $B(n) = 2^{n^c}$ of its inputs, then A finds a satisfying input for C. Then, there exists a polynomial-time algorithm A' that, when given as input a polynomial-size CNF that accepts most of its inputs, then A' finds a satisfying input for the CNF.*

An appealing way to approach this "threshold" at depth four (with $B(n) = 2^{n^c}$) is to derandomize $\mathcal{AC}^0[\oplus]$ *circuits of depth three* with $B(n) = 2^{n^c}$. Goldreich and Wigderson derandomized most types of layered depth-3 $\mathcal{AC}^0[\oplus]$ circuits with $B(n) = 2^{n^c}$, for any $c < 1$, with the exception of circuits of the form $\oplus \wedge \oplus$ (i.e., top $\oplus$ gate, middle layer of $\wedge$ gates, and a bottom layer of $\oplus$ gates), which they left as an open problem.

Our main result in this section is an algorithm that makes significant progress on this problem, by derandomizing $\oplus \wedge \oplus$ circuits with $B(n) = 2^{n^c}$ under various sub-quadratic upper bounds on the circuit size, where some of these bounds refer to each layer separately.

▶ **Theorem 6** (Hitting Biased $\oplus \wedge \oplus$ Circuits). *Let $\epsilon > 0$ be an arbitrary constant. Let $\mathcal{C}$ be the class of circuits of depth three with a top $\oplus$ gate, a middle layer of $\wedge$ gates, and a bottom layer of $\oplus$ gates, such that every $C \in \mathcal{C}$ over n input bits satsifies (at least) one of the following:*

1. *The size of C is $O(n)$.*
2. *The number of $\wedge$-gates is at most $n^{2-\epsilon}$, and the number of $\oplus$-gates is at most $n + n^{\epsilon/2}$.*
3. *The number of $\oplus$-gates is at most $n^{1+\epsilon}$, and the number of $\wedge$-gates is at most $\frac{1}{5} \cdot n^{1-\epsilon}$.*

Then, for some $c = c(\epsilon) > 0$, there exists a polynomial-time algorithm that, when given a circuit $C \in \mathcal{C}$ that accepts all but $B(n) = 2^{n^c}$ of its inputs, outputs a satisfying input for C.

We stress that the algorithm from Theorem 6 makes essential use of the specific circuit C that is given to the algorithm as input. For further details see Section 2.3.

1.4 Polynomials that vanish rarely

We now turn our attention to quantified derandomization of polynomials, and specifically to the problem of constructing hitting-set generators for polynomials $\mathbb{F}_2^n \to \mathbb{F}_2$ that *vanish rarely*. In this setting it is more convenient to work with a normalized badness parameter $b(n) = B(n)/2^n$: For an integer n and a degree bound $d < n$, we want to construct a hitting-set generator (with seed length $O(\log(n))$) for the class of polynomials $p : \mathbb{F}_2^n \to \mathbb{F}_2$ of total degree d that vanish on at most a $b(n)$ fraction of their inputs (i.e., $\Pr_{x \in \mathbb{F}_2^n}[p(x) = 0] \leq b(n)$).

The problem is trivial when $b(n) < 2^{-d}$, since in this case p is constant, and Goldreich and Wigderson solved this problem when $b(n) = O\left(2^{-d}\right)$; we provide an alternative proof of their result in Appendix A. They suggested to try and extend this result to also handle $b(n) = m(n) \cdot 2^{-d}$, where $m(n) = \text{poly}(n)$, and conjectured that such a result would imply a quantified derandomization of $\oplus \wedge \oplus$ circuits of size $m(n)$. [1] We affirm their conjecture, by showing that any sufficiently dense hitting-set generator for degree-d polynomials with $b(n) = m(n) \cdot 2^{-d}$ also hits $\oplus \wedge \oplus$ circuits of size $m(n)$ with $B(n) = \Omega\left(2^n\right)$.

▶ **Theorem 7** (Reducing Hitting $\oplus \wedge \oplus$ Circuits to Hitting Biased Polynomials of Bounded Degree). *Let $\mathcal{C}$ be the class of $\oplus \wedge \oplus$ circuits over n input bits with $m = m(n)$ $\wedge$-gates that accept all but $B(n) = \epsilon \cdot 2^n$ of their inputs, where $m(n) = o(2^n)$ and $\epsilon = \epsilon(n) \leq 1/8$. Let $\mathcal{P}$ be the class of polynomials $\mathbb{F}_2^n \to \mathbb{F}_2$ of degree $d = \lfloor \log(m(n)) + \log(1/\epsilon) \rfloor$ that accept all but a $b(n) = (4 \cdot m(n)) \cdot 2^{-d} = 4 \cdot \epsilon$ fraction of their inputs. Then, any hitting-set generator with density $1/2 + 2 \cdot \epsilon$ for $\mathcal{P}$ is also a hitting-set generator for $\mathcal{C}$.*

Our main focus in the current section is an extension of the problem of hitting polynomials that vanish rarely to *fields larger than* $\mathbb{F}_2$. Specifically, let $\mathbb{F}$ be a finite field of size $|\mathbb{F}| = q \leq \text{poly}(n)$, and let $1 \leq d \leq (q-1) \cdot n$. We consider the problem of constructing hitting-set generators for polynomials $\mathbb{F}^n \to \mathbb{F}$ of degree d that vanish on at most a $b(n)$ fraction of their inputs. Recall that any hitting-set generator for the class of *all* polynomial of total degree d (i.e., regardless of the fraction of inputs on which they vanish) requires a seed of $\log\left(\binom{n+d}{d}\right)$ bits, and that there exists a non-explicit pseudorandom generator for this class with a seed of $O\left(\log\left(\binom{n+d}{d}\right)\right)$ bits. [2] Moreover, for $d = O(1)$ and a sufficiently large q, *explicit* constructions of pseudorandom generators with a seed of $O(\log(n))$ bits are known (see, e.g. [2, 6]).

Our question is whether it is possible to use a *shorter seed* if we only require that the generator will hit degree-d polynomials that vanish on $b(n)$ of their inputs. More accurately, we ask *how low* must $b(n)$ be in order for a hitting-set generator with seed length $o\left(\log\left(\binom{n+d}{d}\right)\right)$ to exist, even non-explicitly. The setting of $b(n) < q^{-d}$ is trivial, since any degree-d polynomial that has at least one root vanishes on at least q^{-d} of its inputs (this follows from Warning's second theorem; see, e.g., [19, Sec. 4]). On the other hand, the

[1] In [9, Sec. 6 (full version)] it is suggested to prove this result by modifying any $\oplus \wedge \oplus$ circuit to a bounded-degree polynomial, where the modification amounts to the removal of all $\wedge$-gates with high fan-in. However, as explained in Section 2.3, since the top gate is a $\oplus$-gate, we cannot simply remove $\wedge$-gates with high fan-in (or remove some of the wires that feed into them).

[2] For proof of the lower bound see, e.g., the proof of Theorem 41, and for the upper bound note that a polynomial $\mathbb{F}^n \to \mathbb{F}$ of degree d can be represented by $\binom{n+d}{d} \cdot \log(q)$ bits.

setting of $b(n) = d/q$ is essentially the standard (i.e., non-quantified) problem, since any non-zero degree-d polynomial vanishes on at most d/q of its inputs.

Our first result for this problem is that for any degree $d \leq 0.99 \cdot q$, any hitting-set generator for degree-d polynomials with $b(n) = O(1/q)$ requires a seed of $\Omega\left(\log\left(\binom{n+d}{d}\right)\right)$ bits; that is, the value $b(n) = O(1/q)$ yields essentially no relaxation *at all* (with respect to seed length), compared to the standard problem. Indeed, *most* polynomials of degree d vanish on at most a $O(1/q)$ fraction of their inputs, but the fact that this is the typical case does not a-priori imply that it is not easier to handle.

Our main result for this problem, however, goes much further: It turns out that even when considering the parameter $b(n) = 1/\text{poly}(q)$, any hitting-set generator for degree-d polynomials that vanish on $b(n)$ of their inputs still requires a seed of length similar to that of a hitting-set generator for *all* degree-d polynomials. Specifically, any hitting-set generator for degree-d polynomials with $b(n) = 1/\text{poly}(q)$ requires a seed of $\Omega\left(\log\left(\binom{n+d^{1/O(1)}}{d^{1/O(1)}}\right)\right)$ bits. It follows that for *any* super-constant degree $d = \omega(1)$, there does not exist a hitting-set generator with seed length $O(\log(n))$ for degree-d polynomials with $b(n) = 1/\text{poly}(q)$.

▶ **Theorem 8** (Hitting Polynomials that Vanish Rarely Over Large Fields; Informal). *For a constant $k \in \mathbb{N}$, let $n \in \mathbb{N}$, and let $\mathbb{F}$ be a field of size $|\mathbb{F}| = q \leq n^k$. Then:*

1. *For any degree $d \leq 0.99 \cdot q$, any hitting-set generator with constant density for the class of polynomials $\mathbb{F}^n \to \mathbb{F}$ of degree d that vanish on at most $b(n) = O(1/q)$ their inputs requires a seed of $\Omega\left(\log\left(\binom{n+d}{d}\right)\right)$ bits.*
2. *For any even constant $t \geq 2$ and degree $d' \leq 0.99 \cdot q^{t+1}$, any hitting-set generator for the class of polynomials $\mathbb{F}^n \to \mathbb{F}$ of degree d' that vanish on at most $b(n) = O\left(q^{-t^2/4}\right)$ of their inputs requires a seed of $\Omega\left(\log\left(\binom{n+d}{d}\right)\right)$ bits, where $d = (d')^{1/(t+1)}$.*

The proofs of both items of Theorem 8 consist of reducing the problem of constructing a hitting-set generator for *all* polynomials of degree $d \in \mathbb{N}$ to the problem of constructing a hitting-set generator for polynomials that *vanish rarely* and are of degree d', where $d' = d$ in the proof of Item (1) and $d' = \text{poly}(d)$ in the proof of Item (2). See Section 2.4 for details.

1.5 Organization of the paper

In Section 2 we explain, in high level, the techniques used to obtain our results. Section 3 contains preliminary definitions and statements of some well-known facts, and in Section 4 we prove two lemmas related to the technique of randomized tests that will be used in the paper. Then, each of the subsequent sections includes proofs for a corresponding section from the introduction: In Section 5 we prove Theorems 1 and 2; in Section 6 we prove Theorems 5 and 6; and in Section 7 we prove Theorems 7 and 8. In Appendix A we provide an alternative proof of [9, Thm. 1.6], and in Appendices B and C we provide proofs for several claims from Sections 5 and 7, respectively.

2 Our Techniques

In this section we give overviews of the proofs of the main theorems for each of the three settings: Theorems 2 and 3 for constant-depth circuits; Theorem 6 for constant-depth circuits with parity gates; and Theorem 8 for polynomials over large fields. Since several of our proofs rely on a common technique, we will begin by describing this technique in general terms (the results that use this technique are Theorems 3 and 7, Item (1) of Theorem 8, and also Theorem 42 in Appendix A).

2.1 A general technique: Randomized tests

Let $G \subseteq \{0,1\}^n$ be a set of *good* objects, and assume that we want to efficiently and deterministically find some $x \in G$. A known technique to do so is to design a *simple deterministic test* $T : \{0,1\}^n \to \{0,1\}$ such that $T(x) = 1$ if and only if $x \in G$. The existence of such a test T is useful, since if T is sufficiently simple such that we are able to construct a hitting-set generator for T, then the generator outputs $x \in G$ with positive probability (because the output distribution of the generator contains $x \in \{0,1\}^n$ such that $T(x) = 1$). Indeed, this approach reduces the task of finding $x \in G$ to the task of designing a test T for G that is *sufficiently simple* such that we are able to construct a hitting-set generator for T.

Intuitively, the *randomized tests technique* is based on the observation that an argument similar to the one above holds also when we replace the deterministic test T by a *distribution* $\mathbf{T}$ *over simple (deterministic) tests* such that, for every fixed $x \in \{0,1\}^n$, it holds that $\mathbf{T}(x)$ computes the indicator function of G, with high probability (say, 0.9). To see this, assume that $\mathbf{T}$ is indeed such a distribution, and let $\mathbf{w}$ be a distribution over $\{0,1\}^n$ that is a hitting-set with density $1-\epsilon$ for every $T \in \mathbf{T}$. Then, on the one hand, $\Pr[\mathbf{T}(\mathbf{w}) = 1] \geq 1 - \epsilon$ (because for every $T \in \mathbf{T}$ it holds that $\Pr[T(\mathbf{w}) = 1] \geq 1 - \epsilon$); and on the other hand, $\Pr[\mathbf{T}(\mathbf{w}) = 0] \geq \Pr[\mathbf{w} \notin G] \cdot \max_{x \notin G}\{\Pr[\mathbf{T}(x) = 0]\}$. Combining the two statements, and recalling that for every $x \notin G$ it holds that $\Pr[\mathbf{T}(x) = 0] \geq 0.9$, it follows that $\Pr[\mathbf{w} \notin G] \leq \epsilon/0.9$, which allows us to deduce that $\mathbf{w}$ contains an object in G.

Indeed, this approach reduces the task of finding $x \in G$ to the tasks of designing a distribution $\mathbf{T}$ over simple tests as above, and of constructing a hitting-set generator with high density for the residual (deterministic) tests $T \in \mathbf{T}$. The main benefit in this approach over the previous one (in which we had a single deterministic test) is that in some cases, *the use of randomness allows us to obtain very simple residual tests, which are simpler than any deterministic test for* G; one appealing example for such a case appears in Section 2.2. We stress that when designing the distribution $\mathbf{T}$ we can be wasteful in the use of randomness, because the existence of $\mathbf{T}$ is only a part of the *analysis*: The actual algorithm for finding $x \in G$ is merely a hitting-set generator (for the residual tests $T \in \mathbf{T}$), whereas only the proof that the generator outputs $x \in G$ relies on the existence of the distribution $\mathbf{T}$.

Two relaxations of the hypotheses for the argument above can immediately be made. First, in our argument we only used the fact that $\mathbf{T}(x) = 0$ with high probability for every $x \notin G$ (and did not explicitly rely on the hypothesis that $\mathbf{T}(x) = 1$ with high probability for *every* $x \in G$). And secondly, we do not have to assume that $\mathbf{w}$ is a hitting-set with high density for every $T \in \mathbf{T}$, but rather only need the hypothesis that $\Pr[\mathbf{T}(\mathbf{w}) = 1]$ is high.

Let us demonstrate one appealing setting in which the two relaxed hypotheses above hold, which simplifies and abstracts the setting in the proof of Theorem 3. Assume that there exists a set $E \subseteq G$ of *excellent* objects, and that almost all objects are excellent; that is, a random $x \in \{0,1\}^n$ is not only good, but also has additional useful properties. Also assume that we are able to construct a distribution $\mathbf{T}$ over simple tests that distinguishes between excellent objects and bad ones (i.e., $\mathbf{T}$ solves a promise problem with some "gap" between the "yes" instances and the "no" instances). Denoting the uniform distribution over $\{0,1\}^n$ by $\mathbf{u}_n$, in this case we have that $\Pr[\mathbf{T}(\mathbf{u}_n) = 1]$ is high, whereas $\Pr[\mathbf{T}(x) = 0]$ is high for every $x \notin G$. Indeed, in such a setting, in order to find $x \in G$ it suffices to construct a pseudorandom generator for the residual tests $T \in \mathbf{T}$ (see Lemma 15).

2.2 Constant-depth circuits: Overview of the proofs of Theorems 2 and 3

Theorem 2 is a special case of the more general Theorem 3. However, since there is a simple and more direct way to prove Theorem 2, we describe this simpler way first, and only then turn to the describe the proof of the more general theorem.

Let C be a depth-D circuit that accepts all but $B(n) = \Omega\left(2^{n/\log^{D-2}(n)}\right)$ of its inputs. The hitting-set generator first uses pseudorandom restrictions to simplify C to a depth-2 circuit, by fixing values for all but $n' = \Omega(n/\log^{D-2}(n))$ of the variables. These pseudorandom restrictions are chosen using an adaptation of the derandomized switching lemma of Trevisan and Xue [23] (either Tal's [20] improvement or the adapted version in Proposition 26), which requires a seed of length $\tilde{O}(\log^3(n))$. At this point, there are $n' \geq \log(B(n)) + 1$ living variables, and therefore the simplified circuit (over n' input bits) has acceptance probability at least $1/2$ (since C has at most $B(n)$ unsatisfying inputs). Hence, we can use any pseudorandom generator for depth-2 circuits with seed length at most $\tilde{O}(\log^3(n))$ (e.g., that of De *et al.* [7]) in order to fix values for the remaining n' variables, thus finding a satisfying input for C, with high probability. [3]

Turning to the more general Theorem 3, the high-level structure of its proof is similar to that of the proof of Theorem 2: We first use a derandomized switching lemma to radically simplify the circuit, while keeping more than $\log(B(n))$ variables alive, and then use a pseudorandom generator for the simplified circuit to find a satisfying input. The key difference from Theorem 2 is that the first step uses a new derandomization of the switching lemma, which we establish.

The new derandomization of the switching lemma depends on the *width* (i.e., bottom fan-in) of the depth-2 formula that we want the restriction to simplify. Previous known derandomizations of the lemma can also be adapted to depend on the width of the formula: For typical settings of the parameters (e.g., polynomially-small error), the derandomization of Goldreich and Wigderson [9] can be adapted to yield a seed length of $\tilde{O}(2^w) \cdot \log(n)$ for formulas of width w (see Proposition 44), and the derandomization of Trevisan and Xue [23] can be adapted (using the pseudorandom generator of Gopalan, Meka, and Reingold [10]) to yield a seed length of $\tilde{O}(w) \cdot \log^2(n)$ (see Proposition 26). We show a derandomization that requires a seed of length $\tilde{O}(w^2 \cdot \log(n))$ (see Proposition 28). Indeed, in this new result, the dependency of the seed length on w is exponentially better than in [9], and the seed length is shorter than in [23] for any $w = o(\log(n))$. The caveat, however, is that we do not show that the formula itself is simplified in the subcube corresponding to the restriction; instead, we show that the formula is *approximated* by a decision tree of bounded depth in this subcube (i.e., there exists such a decision tree that agrees with the formula on almost all inputs in the subcube). This weaker conclusion suffices for our main application (i.e., for Theorem 3) as well as for all other applications of derandomized switch lemmas that we are aware of.

Our starting point in the proof of this lemma is a result of Gopalan, Meka, and Reingold [10], which asserts that for any depth-2 formula F of width w and any $\beta > 0$, there exists a formula $F^{\texttt{low}}$ of width at most w and size at most $m' = 2^{\tilde{O}(w)\cdot\log\log(1/\beta)}$ such that $F^{\texttt{low}}$ is "lower-sandwiching" for F (i.e., $F^{\texttt{low}}(x) \leq F(x)$ for all $x \in \{0,1\}^n$) and

[3] Actually, there is one minor subtlety in this description: In the derandomizations of [23, 20], the expected number of living variables is close to $n/\log^{d-2}(n)$, but it is *not* guaranteed that approximately this many variables remain alive with high (or even constant) probability. Nevertheless, the latter does hold when instantiating their generic construction in a specific manner; see the proof of Theorem 3 for further details.

$\Pr_{x\in\{0,1\}^n}[F(x) \neq F^{\mathtt{low}}(x)] \leq \beta$. Now, since $F^{\mathtt{low}}$ is both small (i.e., m' is upper bounded) and of bounded width, we can find a restriction that simplifies it using a relatively short seed; specifically, we can use an adapted version of the lemma of [23] (see Proposition 26), and the required seed length (when we want the probability of error to be $1/\mathrm{poly}(n)$) is only $\tilde{O}(w) \cdot \log(m') \cdot \log(n) = \tilde{O}(w^2) \cdot \log(n) \cdot \log\log(1/\beta)$.

The main challenge that underlies this approach is that, while $F^{\mathtt{low}}$ agrees with F on most inputs $x \in \{0,1\}^n$, it is not clear that $F^{\mathtt{low}}$ also agrees with F on most inputs *in the subcube that corresponds to* ρ; that is, it is not guaranteed that $F^{\mathtt{low}}{\restriction_\rho}$ will agree with $F{\restriction_\rho}$ on most of *their* inputs. To make sure that $F^{\mathtt{low}}{\restriction_\rho}$ will agree with $F{\restriction_\rho}$ on most of their inputs, we will choose ρ such that it "fools" additional tests that check whether or not $F^{\mathtt{low}}{\restriction_\rho}$ and $F{\restriction_\rho}$ indeed typically agree. To design these tests we use the randomized tests technique: Specifically, a natural randomized test to decide whether or not $F^{\mathtt{low}}{\restriction_\rho}$ and $F{\restriction_\rho}$ typically agree is to sample random inputs inside the subcube that corresponds to ρ, and accept if and only if $F^{\mathtt{low}}{\restriction_\rho}$ and $F{\restriction_\rho}$ agree on the sampled inputs.

Indeed, *the residual tests under this distribution are simpler (in any reasonable sense) than any deterministic test* that decides whether or not $F^{\mathtt{low}}{\restriction_\rho}$ and $F{\restriction_\rho}$ agree on most of their inputs. The remaining task is thus to construct a hitting-set generator with high density for these residual tests. We will now describe how to do so, relying both on the specific details of the construction of $F^{\mathtt{low}}$ from [10], in order to construct circuits with a specific structure that will be convenient for us for each residual test, and on relaxations of the randomized tests technique that follow the ones suggested in the end of Section 2.1.

We want to use the lemma to simplify polynomially-many depth-2 formulas (i.e., simplify an entire "layer" of a constant-depth circuit). Thus, we want that for every fixed formula F it will hold that $F^{\mathtt{low}}{\restriction_\rho}$ and $F{\restriction_\rho}$ agree on an all but an α-fraction of their inputs, where $\alpha = 1/\mathrm{poly}(n)$. We say that a restriction ρ is *good* if $F^{\mathtt{low}}{\restriction_\rho}$ and $F{\restriction_\rho}$ agree with probability at least $1-\alpha$. If we start from a formula $F^{\mathtt{low}}$ with the approximation parameter $\beta = \mathrm{poly}(\alpha)$, then almost all restrictions ρ' are *excellent*, in the sense that $F^{\mathtt{low}}{\restriction_{\rho'}}$ and $F{\restriction_{\rho'}}$ agree with probability $1-\sqrt{\beta} \gg 1-\alpha$. For each fixed F and $F^{\mathtt{low}}$, to distinguish between excellent restrictions and restrictions that are not good, the distribution $\mathbf{T}$ of tests uniformly samples $\mathrm{poly}(\alpha)$ inputs inside the subcube that corresponds to its input restriction ρ, and accepts ρ if and only if F and $F^{\mathtt{low}}$ agree on the sampled inputs.

The next step is to show that each residual test $T \in \mathbf{T}$ can be computed by a circuit with a convenient structure. To do so, we observe that the construction of $F^{\mathtt{low}}$ in [10] is based on a sequence of *specific syntactic modifications* to F: Each syntactic modification is a simplification of a quasi-sunflower, a notion introduced by Rossman [18] (for more specific details see Section 5.2.1). We define the tests $T \in \mathbf{T}$ to accept if and only if the *specific syntactic modifications* used to transform F into $F^{\mathtt{low}}$ did not affect the formula at the relevant inputs. Then, we show that each such test T can be decided by a depth-3 circuit with a top AND gate and bottom fan-in w (relying on the hypothesis that the original formula F has width w; see Claim 29.3).

Now, since almost all restrictions are excellent, and each excellent restriction is accepted with high probability by $\mathbf{T}$, it follows that almost all tests in $\mathbf{T}$ belong to the subset $\mathbf{T}' \subseteq \mathbf{T}$ of tests that accept almost all of their input restrictions. We will in fact construct a hitting-set generator for the residual tests $T \in \mathbf{T}'$. This can be done relying both on the fact that $T \in \mathbf{T}'$ has very high acceptance probability and on the fact that it can be computed by a depth-3 circuit with a top AND gate and bottom fan-in w (the latter allows us to use the pseudorandom generator of [10] for formulas of small width; see Claim 29.4).

To prove Theorem 3, we will repeat the following step: First reduce the width of the formulas in the next-to-bottom layer by a pseudorandom restriction (see Claim 30.1), and

then use the new switching lemma to approximate the circuit by a circuit in which all the formulas in the next-to-bottom layer are simplified (and thus the latter circuit has smaller depth). Since all our approximations are "lower-sandwiching", any satisfying input for the latter circuit is also satisfying for the former circuit.

2.3 Constant-depth circuits with parity gates: Overview of the proof of Theorem 6

Let us now describe the high-level strategy of the algorithms of Theorem 6. First observe that any $\oplus \wedge \oplus$ circuit C computes an n-variate polynomial over $\mathbb{F}_2$, and that the total degree of this polynomial equals the maximal fan-in of $\wedge$-gates in the circuit. Our approach will be to find an *affine subspace* W of dimension more than $\log(B(n))$ such that when C is restricted to the affine subspace, the fan-in of all $\wedge$-gates becomes constant. Thus, when restricted to W, the circuit C becomes a non-zero polynomial of constant degree, which means that we can then hit it using a pseudorandom generator for polynomials of constant degree (i.e., Viola's [25]).

In order to find the affine subspace W, we use *affine restrictions*, which are obtained by fixing values to some of the bottom $\oplus$-gates. These are analogous to standard "bit-fixing" restrictions, but in contrast to the latter, we cannot consider *any* sequence of fixed values to the bottom $\oplus$-gates: This is the case because the bottom $\oplus$-gates might not be linearly independent (and thus the values of some $\oplus$-gates might depend on the values of other $\oplus$-gates). In particular, this means that we cannot use random (or pseudorandom) restrictions in which the value of each $\oplus$-gate is chosen obliviously of the $\oplus$-gates of the circuit.

Our algorithm circumvents this problem by constructing a restriction that corresponds to the *specific* $\oplus \wedge \oplus$ circuit that is given to the algorithm as input. For concreteness, let us now describe the construction of Item (2) of Theorem 6, and let us also fix specific parameter values to work with: We assume, for simplicity, that the number of bottom $\oplus$-gates is *exactly* n; and we assume that the number of $\wedge$-gates is $n^{1.1}$, and that the circuit accepts all but $\Omega\left(2^{n^{1/3}}\right)$ of its inputs.

First assume, for a moment, that the fan-in of each $\wedge$-gate in the middle layer of the circuit is upper bounded by $\sqrt{n}$. In this case we can restrict the $\oplus$-gates as follows. Consider a random restriction process in which each bottom $\oplus$-gate is fixed independently with probability $1 - p = 1 - n^{-2/3}$, and the *values* for the fixed gates are chosen afterwards, in an *arbitrary consistent manner*. With high probability, the restriction will yield a subspace of dimension approximately $p \cdot n = n^{1/3} > \log(B(n))$. Also, since each $\wedge$-gate g has fan-in at most $w = \sqrt{n}$, and $p = 1/w^{1+\Omega(1)}$, with high probability, all but $O(1)$ of the gates that feed into g are fixed by this process.[4] In fact, the above two statements hold even if we choose the restriction according to an $O(1)$-independent distribution, rather than uniformly.

Needless to say, we cannot actually assume that the fan-in of $\wedge$-gates is bounded by $\sqrt{n}$. Thus, our strategy will be to first *mildly* reduce the fan-in of $\wedge$-gates (from n to $\sqrt{n}$), and then invoke the restriction process described above. A standard approach to mildly reduce the fan-in of $\wedge$-gates is to simply remove some of the incoming wires to each $\wedge$-gate. However, this approach *does not* work in our setting, since the top gate is a $\oplus$-gate, which means that such a modification might turn unsatisfying inputs into satisfying ones (and thus hitting the modified circuit might not yield a satisfying input to the original circuit).

[4] For any $\wedge$-gate g with initial fan-in $d_\wedge$, the probability that there exists a set of size c of $\oplus$-gates that feed into g that are all unfixed is at most $\binom{d_\wedge}{c} \cdot p^c = 1/\mathrm{poly}(n)$, for a sufficiently large $c = O(1)$.

To reduce the fan-in of $\wedge$-gates to $\sqrt{n}$, we follow Kopparty and Srinivasan [13] in adapting the approach of Chaudhuri and Radhakrishnan [4] to the setting of $\oplus\wedge\oplus$ circuits. [5] Specifically, we first iteratively fix each $\oplus$-gate that has *fan-out* more than $n^{1/4}$ to a *non-accepting* value; note that such an action also fixes $n^{1/4}$ $\wedge$-gates in the middle layer, and hence in this step we fix values for at most $n^{1.1}/n^{1/4} = o(n)$ bottom $\oplus$-gates (because afterwards there are no more living $\wedge$-gates). At this point, the number of wires feeding the middle layer is at most $n \cdot n^{1/4} = n^{1.25}$. Now, for each $\wedge$-gate g with *fan-in* more than $\sqrt{n}$, we fix a $\oplus$-gate that feeds into g to a *non-accepting* value, thereby also fixing g; each such action eliminates $\sqrt{n}$ wires that feed into the middle layer, and therefore in this step we fix at most $n^{1.25}/\sqrt{n} = o(n)$ bottom $\oplus$-gates. Overall, the fan-in of each $\wedge$-gate has been reduced to $\sqrt{n}$, and we imposed at most $o(n)$ affine conditions.

To see that the final subspace W is of dimension more than $\log(B(n))$, note that the dimension of W equals the number of living $\oplus$-gates (because we assumed that the initial number of $\oplus$-gates is exactly n). After the first step of the algorithm (i.e., reducing the fan-in of $\wedge$-gates to $\sqrt{n}$), we are left with $(1 - o(1)) \cdot n$ living $\oplus$-gates, and the second step (i.e., the pseudorandom restriction) leaves a fraction of $p = n^{-2/3}$ of them alive. Thus, the expected dimension of W is $\Omega(p \cdot n) = \Omega\left(n^{1/3}\right) > \log(B(n))$.

The approach above actually works for a broader range of parameters, and in particular when the number of $\wedge$-gates is $n^{2-\epsilon}$, for any constant $\epsilon > 0$, and when the number of $\oplus$-gates is $n + n^c$, for any $c < \epsilon$ (see details in Section 6.2.3). In Items (1) and (3), we consider circuits in which the number of $\oplus$-gates is significantly larger than n, namely $O(n)$ and $O\left(n^{1+\epsilon}\right)$, respectively. The proofs of both these items use algorithms that are variations of the first step of the algorithm described above, and these proofs are detailed in Sections 6.2.2 and 6.2.4, respectively.

2.4 Polynomials that vanish rarely: Overview of the proof of Theorem 8

The main component in the proof of Theorem 8 is a reduction of the task of constructing a hitting-set generator for polynomials $\mathbb{F}^n \to \mathbb{F}$ of degree $d \leq 0.99 \cdot |\mathbb{F}|$ to the task of constructing a hitting-set generator for polynomials $\mathbb{F}^{O(n)} \to \mathbb{F}$ of degree $d' \geq d$ that vanish rarely. Since any hitting-set generator for all polynomials of degree d requires a seed of $\Omega\left(\log\left(\binom{n+d}{d}\right)\right)$ bits, we obtain the lower bound on hitting-set generators for polynomials $\mathbb{F}^{O(n)} \to \mathbb{F}$ of degree d' that vanish rarely. The aforementioned reduction can be thought of as a form of "randomness-efficient error reduction" for polynomials such that the increase in degree from d to d' is mild (or even $d' = d$).

Let $p : \mathbb{F}^n \to \mathbb{F}$ be of degree d. The first observation is that since $d \leq 0.99 \cdot |\mathbb{F}|$, it holds that $\Pr_{x \in \mathbb{F}^n}[p(x) = 0] \leq 0.99$, which implies that the probability over a random subspace $W \subseteq \mathbb{F}^n$ of constant dimension that $p\restriction_W \equiv 0$ is very small (because such a subspace consists of poly($|\mathbb{F}|$) points that are $O(1)$-wise independent). Our strategy will be to try and construct a polynomial $p' : \mathbb{F}^{O(n)} \to \mathbb{F}$ that satisfies the following: The polynomial p' gets as input a tuple $\vec{u} \in \mathbb{F}^{O(n)}$ that defines a subspace $W = W_{\vec{u}}$, and outputs zero if and only if $p\restriction_W \equiv 0$. Note that any polynomial p' that satisfies this condition vanishes rarely, because $p\restriction_W \not\equiv 0$

[5] Originally, [4] applied their approach to $\mathcal{AC}^0$ circuits, and [13] later adapted this approach to $\mathcal{AC}^0[\oplus]$ circuits. Our adaptation is slightly different technically than in [13], to suit the specific circuit structure $\oplus \wedge \oplus$; but more importantly, while both [4, 13] use the approach as part of the analysis (to prove lower bounds), we use this approach as a (non-black-box) algorithm for derandomization.

for almost all subspaces W. And indeed, hitting p' yields a subspace W such that $p\restriction_W \not\equiv 0$, which allows us to hit p, by using additional $O(\log(|\mathbb{F}|)) \leq O(\log(n))$ random bits to choose $w \in W$. (This approach is reminiscent of Bogdanov's [2] reduction of the construction of pseudorandom generators to the construction of hitting-set generators.)

The main challenge in constructing such a polynomial p' is the following: Given a tuple $\vec{u} \in \mathbb{F}^{O(n)}$ that defines a subspace $W = W_{\vec{u}} \subseteq \mathbb{F}^n$, how can we test efficiently (i.e., with degree d' that is not much larger than d) whether or not $p\restriction_W \not\equiv 0$? Indeed, a naive solution is to compute the OR function of the values $\{p(w) : w \in W\}$ (i.e., compute the polynomial that outputs 1 if and only if there exists $w \in W$ such that $p(w) \neq 0$), but this solution requires a very high degree $d' \geq \text{poly}(|\mathbb{F}|)$. We present two solutions for this problem: The first yields $d' = \text{poly}(d)$, and corresponds to Item (2) of Theorem 8, and the second yields $d' = d$, and corresponds to Item (1) of Theorem 8.

The first solution relies on the observation that instead of testing whether or not there exists $w \in W$ such that $p(w) \neq 0$, we can test whether or not there exists a non-zero coefficient in the representation of $p\restriction_W$ as a polynomial $\mathbb{F}^{O(1)} \to \mathbb{F}$. Since $p\restriction_W$ is of degree d, the number of coefficients of $p\restriction_W$ is $\text{poly}(d)$. Moreover, each of the coefficients of $p\restriction_W$ is actually a polynomial of degree d in $\vec{u}$ (see Claim 39.1). Thus, instead of taking an OR of $\text{poly}(|\mathbb{F}|)$ values (i.e., of the values in $\{p(w) : w \in W\}$), we can take an OR of $\text{poly}(d)$ values, where each of these values can be computed by a polynomial of degree d in $\vec{u}$.

The first solution is not complete yet, since computing the OR function of $k = \text{poly}(d)$ values requires degree $(|\mathbb{F}| - 1) \cdot k$. To solve this problem, observe that we do not actually need to output 1 on every non-zero input; in fact, it suffices that on every non-zero input, we output *some* non-zero value in $\mathbb{F}$. We call such functions multivalued OR functions, and show that there exists a polynomial $\mathbb{F}^k \to \mathbb{F}$ of degree less than $2 \cdot k$ that computes a multivalued OR function of its inputs (see Proposition 38). It follows that there exists a polynomial $p' : \mathbb{F}^{O(n)} \to \mathbb{F}$ of degree $d' = \text{poly}(d)$ that vanishes on at most $1/\text{poly}(|\mathbb{F}|)$ of its inputs (corresponding to the probability that $p\restriction_W \equiv 0$) such that every non-zero input $\vec{u}$ to p' yields a subspace $W = W_{\vec{u}}$ such that $p\restriction_W \not\equiv 0$.

The solution described above yields the lower bound in Item (2) of Theorem 8, which refers to the badness parameter $b(n) = 1/\text{poly}(|\mathbb{F}|)$. To obtain the lower bound in Item (1), we will again reduce the task of hitting $p : \mathbb{F}^n \to \mathbb{F}$ to the task of finding a subspace W such that $p\restriction_W \not\equiv 0$, but we will then further reduce the latter task to the task of hitting polynomials of degree d that vanish on at most $O(1/|\mathbb{F}|)$ of their inputs. To do so, we use a variation on the technique of randomized tests. Specifically, we construct a distribution $\mathbf{h}$ over polynomials $\mathbb{F}^{O(n)} \to \mathbb{F}$ that satisfies: (1) For every $\vec{u} \in \mathbb{F}^{O(n)}$ such that $p\restriction_{W_{\vec{u}}} \equiv 0$ it holds that $\mathbf{h}(\vec{u}) = 0$, with probability one; (2) The distribution $\mathbf{h}$ is typically in the class $\mathcal{P}$ of degree-d polynomials that vanish on at most $O(1/|\mathbb{F}|)$ of their inputs. We will then rely on arguments similar to those in Section 2.1, to deduce that any sufficiently dense hitting-set generator for $\mathcal{P}$ outputs $\vec{u}$ such that $p\restriction_{W_{\vec{u}}} \not\equiv 0$ (see Lemma 16).

Recall that the coefficients of $p\restriction_{W_{\vec{u}}}$ are degree-d polynomials in $\vec{u}$. The aforementioned distribution, denoted by $\mathbf{h}$, is simply a random $\mathbb{F}$-linear combination of these degree-d polynomials. Note that $\mathbf{h}$ is supported on polynomials of degree d, and indeed for every $\vec{u}$ such that $p\restriction_{W_{\vec{u}}} \equiv 0$ it holds that $\mathbf{h}(\vec{u}) = 0$, with probability one. Moreover, since almost all $\vec{u}$'s are such that $p\restriction_{W_{\vec{u}}} \not\equiv 0$, and for each such $\vec{u}$ it holds that $\Pr[\mathbf{h}(\vec{u}) \neq 0] = 1 - 1/|\mathbb{F}|$, the expected fraction of inputs on which a polynomial in $\mathbf{h}$ vanishes is at most $O(1/|\mathbb{F}|)$. Thus, most of the polynomials in the support of $\mathbf{h}$ are in $\mathcal{P}$. We can therefore deduce that any sufficiently dense hitting-set generator for $\mathcal{P}$ also outputs $\vec{u}$ such that $p\restriction_{W_{\vec{u}}} \not\equiv 0$, which allows us to hit p using additional $O(\log(|\mathbb{F}|)) = O(\log(n))$ bits.

3 Preliminaries

Throughout the paper, the letter n will always denote the number of input variables to a function or a circuit. We denote by $\{\mathfrak{D} \to \mathfrak{R}\}$ the set of functions from domain $\mathfrak{D}$ to range $\mathfrak{R}$. Distributions and random variables will always be denoted by boldface letters. Given a set Σ, which will typically be clear from the context, we denote by $\mathbf{u}_k$ the uniform distribution over Σ^k. Given a distribution $\mathbf{d}$, we write $x \sim \mathbf{d}$ to denote a value x that is sampled according to $\mathbf{d}$; when we write $x \in \Sigma^k$ in probabilistic expressions, we mean the uniform distribution over Σ^k.

3.1 Circuit classes and restrictions

We will consider Boolean circuit families $\{C_n\}_{n\in\mathbb{N}}$ such that C_n gets n input bits and outputs a single bit. The circuit class $\mathcal{AC}^0$ consists of all circuit families over the De-Morgan basis (i.e., the gates of the circuit can compute the $\wedge, \vee$, and $\neg$ functions) such that the circuit gates have unbounded fan-in and fan-out, and for every $n \in \mathbb{N}$, the size of C_n (i.e., number of gates) is at most $\text{poly}(n)$, and the depth of C_n (i.e., longest path from an input gate to the output gate) is upper bounded by a constant. We also assume that for every $n \in \mathbb{N}$ it holds that C_n has $2 \cdot n$ input gates that correspond to the input literals (i.e., the input bits $x_1, ..., x_n$ and their negations $\neg x_1, ..., \neg x_n$); and that C_n is *layered*, in the sense that in a fixed circuit, for every integer d, all gates at distance d from the input gates are of the same gate-type (i.e., either $\wedge$ or $\vee$).

The circuit class $\mathcal{AC}^0[\oplus]$ is defined similarly to $\mathcal{AC}^0$, the only difference being that the basis is extended: The gates can compute the $\wedge, \vee, \neg$, and $\oplus$ functions (rather than only $\wedge, \vee$, and $\neg$). We stress that a $\oplus$-gate can compute either the parity of its input gates, or the negated parity of its input gates. We also assume that all $\mathcal{AC}^0[\oplus]$ circuits are *layered*, in the sense that in a fixed circuit, for every integer d, all gates at distance d from the input gates are of the same gate-type (i.e., either $\wedge$, or $\vee$, or $\oplus$).

Given a function $f : \{0,1\}^n \to \{0,1\}$, a restriction of f is a subset $W \subseteq \{0,1\}^n$. We say that a function f simplifies under a restriction W to a function from a class $\mathcal{H}$ if there exists $h \in \mathcal{H}$ such that for every $w \in W$ it holds that $h(w) = f(w)$. A restriction to a subcube is represented by a string $\rho \in \{0,1,\star\}^n$, where the subcube consists of all $x \in \{0,1\}^n$ such that for every $i \in [n]$ for which $\rho_i \neq \star$ it holds that $x_i = \rho_i$. The living variables under ρ are the input bits indexed by the set $\{i \in [n] : \rho_i = \star\}$. The restricted function $f\restriction_\rho : \{0,1\}^n \to \{0,1\}$ is defined by $f\restriction_\rho(x) = f(y)$, where for every $i \in [n]$ it holds that $y_i = x_i$ if $\rho_i = \star$ and $y_i = \rho_i$ otherwise. We will also consider the composition of restrictions, where a composition $\rho = \rho_1 \circ \rho_2$ yields the restricted function $f\restriction_\rho = \left(f\restriction_{\rho_2}\right)\restriction_{\rho_1}$.

3.2 Pseudorandom generators and hitting-set generators

We will use the following two standard definitions of pseudorandom generators and of hitting-set generators.

▶ **Definition 9** (Pseudorandom Generators). Let $\mathcal{F} = \bigcup_{n\in\mathbb{N}} \mathcal{F}_n$, where for every $n \in \mathbb{N}$ it holds that $\mathcal{F}_n$ is a set of functions $\{0,1\}^n \to \{0,1\}$, and let $\epsilon : \mathbb{N} \to [0,1]$ and $\ell : \mathbb{N} \to \mathbb{N}$. An algorithm G is a pseudorandom generator for $\mathcal{F}$ with error parameter ϵ and seed length ℓ if for every $n \in \mathbb{N}$, when G is given as input 1^n and a random seed of length $\ell(n)$, it outputs a string in $\{0,1\}^n$ such that for every $f \in \mathcal{F}_n$ it holds that $\left|\Pr_{x\in\{0,1\}^n}[f(x) = 1] - \Pr_{y\in\{0,1\}^{\ell(n)}}[f(G(1^n, y)) = 1]\right| < \epsilon$.

If G is a pseudorandom generator with error parameter ϵ for a class of functions $\mathcal{F}$, then we say that functions from $\mathcal{F}$ are ϵ-fooled by G.

▶ **Definition 10** (Hitting-Set Generators). Let $\mathcal{F} = \bigcup_{n\in\mathbb{N}} \mathcal{F}_n$, where for every $n \in \mathbb{N}$ it holds that $\mathcal{F}_n$ is a set of functions $\{0,1\}^n \to \{0,1\}$, and let $\ell : \mathbb{N} \to \mathbb{N}$. An algorithm G is a hitting-set generator for $\mathcal{F}$ with seed length ℓ if for every $n \in \mathbb{N}$, when G is given as input 1^n and a random seed of length $\ell(n)$, it outputs a string in $\{0,1\}^n$ such that for every $f \in \mathcal{F}_n$ it holds that $\Pr_{y\in\{0,1\}^{\ell(n)}}[f(G(1^n, y)) \neq 0] > 0$. For $\epsilon : \mathbb{N} \to (0, 1]$, we say that G has density ϵ if for every $n \in \mathbb{N}$ and $f \in \mathcal{F}_n$ it holds that $\Pr_{y\in\{0,1\}^{\ell(n)}}[f(G(1^n, y)) \neq 0] \geq \epsilon(n)$.

We now extend Definition 10 by defining hitting-set generators for functions over fields larger than $\mathbb{F}_2$. The following definition requires that the generator G will output a value x such that the relevant function evaluates to any non-zero value on x.

▶ **Definition 11** (Hitting-Set Generators Over Large Fields). For every $n \in \mathbb{N}$, let $\mathbb{F}$ be a finite field of size that may depend on n, and let $\mathcal{F}_n$ be a set of functions $\mathbb{F}^n \to \mathbb{F}$. Let $\mathcal{F} = \bigcup_{n\in\mathbb{N}} \mathcal{F}_n$. For a function $\ell : \mathbb{N} \to \mathbb{N}$, an algorithm G is a hitting-set generator for $\mathcal{F}$ with seed length ℓ if for every $n \in \mathbb{N}$, when G is given as input 1^n and a random seed of $\ell(n)$ bits (i.e., a random string in $\{0,1\}^{\ell(n)}$), it outputs n elements of $\mathbb{F}$ such that for every $f \in \mathcal{F}_n$ it holds that $\Pr_{y\in\{0,1\}^{\ell(n)}}[f(G(1^n, y)) \neq 0] > 0$. For $\epsilon : \mathbb{N} \to (0, 1]$, we say that G has density ϵ if for every $n \in \mathbb{N}$ and $f \in \mathcal{F}_n$ it holds that $\Pr_y[f(G(1^n, y)) \neq 0] \geq \epsilon(n)$.

In Definition 11, the generator G gets a seed from $\{0,1\}^\ell$, rather than from $\mathbb{F}^\ell$ (as is also common in some texts); indeed, the seed length $\ell(n)$ of the generator G might depend on the size of $\mathbb{F}$. This choice was made because it is more general, and because we want to measure the seed length in bits.

3.3 Distributions with limited independence

We say that random variables $\mathbf{x}_1, ..., \mathbf{x}_n \in \{0,1\}^n$ are t-wise independent if for every set $S \subseteq [n]$ of size $|S| = t$, the marginal distribution $(\mathbf{x}_i)_{i\in S}$ is uniform over $\{0,1\}^t$. We will use the following well-known tail bound (for a proof see [1, Lemma 2.3]):

▶ **Fact 12** (Tail Bound for t-Wise Independent Distributions). *Let $n \in \mathbb{N}$, and let $t \geq 4$ be an even number. Let $\boldsymbol{x}_1, ..., \boldsymbol{x}_n$ be random variables in $\{0,1\}$ that are t-wise independent, and denote $\mu = \mathbb{E}\left[\frac{1}{n} \cdot \sum_{i\in[n]} \boldsymbol{x}_i\right]$. Then, for any $\zeta > 0$ it holds that $\Pr\left[\left|\frac{1}{n} \cdot \sum_{i\in[n]} \boldsymbol{x}_i - \mu\right| \geq \zeta\right] \leq 8 \cdot \left(\frac{t\cdot\mu\cdot n+t^2}{\zeta^2\cdot n^2}\right)^{t/2}$.*

We say that $\mathbf{x}_1, ..., \mathbf{x}_n \in \{0,1\}^n$ are δ-almost t-wise independent if for every set $S \subseteq [n]$ of size $|S| = t$, the statistical distance between $(\mathbf{x}_i)_{i\in S}$ and the uniform distribution over $\{0,1\}^t$ is at most δ. Then, the following well-known tail bound holds:

▶ **Fact 13** (Tail Bound for Almost t-Wise Independent Distributions). *Let $n \in \mathbb{N}$, let $t \geq 4$ be an even number, and let $\delta > 0$. Let $\boldsymbol{x}_1, ..., \boldsymbol{x}_n$ be random variables in $\{0,1\}$ that are δ-almost t-wise independent, and denote $\mu = \mathbb{E}\left[\frac{1}{n} \cdot \sum_{i\in[n]} \boldsymbol{x}_i\right]$. Then, for any $\zeta > 0$ it holds that $\Pr\left[\left|\frac{1}{n} \cdot \sum_{i\in[n]} \boldsymbol{x}_i - \mu\right| \geq \zeta\right] < 8 \cdot \left(\frac{t\cdot\mu\cdot n+t^2}{\zeta^2\cdot n^2}\right)^{t/2} + (2 \cdot n)^t \cdot \delta$.*

For a proof of Fact 13 see, e.g., [14, Lemma 18]. We will frequently use Fact 13 with the parameters $t = O(1)$, and $\zeta = \mu/2$, and $\delta = 1/p(n)$ where p is a sufficiently large polynomial; in this case, we have that $\Pr\left[\frac{1}{n} \cdot \sum_{i\in[n]} \mathbf{x}_i \notin \mu \pm (\mu/2)\right] = O\left(1/\left(\mu \cdot n\right)^{t/2}\right)$.

We will also need the following fact, which, loosely speaking, asserts that concatenating two independently-chosen distributions that are almost t-wise independent yields a distribution that is still almost t-wise independent.

▶ **Fact 14** (Concatenating Almost t-Wise Independent Distributions). *Let $n, n' \in \mathbb{N}$, let $\delta, \delta' < \frac{1}{2}$, and let $t \in \mathbb{N}$. Let $\boldsymbol{y}$ be a distribution over $\{0,1\}^n$ that is δ-almost t-wise independent, and let $\boldsymbol{z}$ be a distribution over $\{0,1\}^{n'}$ that is δ'-almost t-wise independent. Let $\boldsymbol{r} = \boldsymbol{y} \circ \boldsymbol{z}$ be a distribution that is obtained by concatenating a sample from $\boldsymbol{y}$ and an independent sample from $\boldsymbol{z}$. Then, the distribution $\boldsymbol{r}$ is $(\delta + \delta')$-almost t-wise independent.*

Proof. Fix a set $S \subseteq [n + n']$ of size $|S| = t$, and let us prove that the ℓ_1-distance between $\mathbf{r}_S$ and the uniform distribution is at most $2 \cdot (\delta + \delta')$ (which implies that the statistical distance between them is at most $\delta + \delta'$). Partition S into $W = S \cap [n]$ and $W' = S \setminus [n]$, and denote $w = |W|$ and $w' = |W'|$. Then, we have that:

$$\begin{aligned}\|\mathbf{r}_S - \mathbf{u}_t\|_1 &= \|\mathbf{y}_W \circ \mathbf{z}_{W'} - \mathbf{u}_w \circ \mathbf{u}_{w'}\|_1 \\ &\leq \|\mathbf{y}_W \circ \mathbf{z}_{W'} - \mathbf{y}_W \circ \mathbf{u}_{w'}\|_1 + \|\mathbf{y}_W \circ \mathbf{u}_{w'} - \mathbf{u}_w \circ \mathbf{u}_{w'}\|_1 \\ &= \|\mathbf{z}_{W'} - \mathbf{u}_{w'}\|_1 + \|\mathbf{y}_W - \mathbf{u}_w\|_1 \ ,\end{aligned}$$

which is upper-bounded by $2 \cdot \delta' + 2 \cdot \delta$. ◀

4 Randomized tests

In this section we state and prove three lemmas that are related to the technique of randomized tests. The first lemma (i.e., Lemma 15) corresponds to the high-level description in Sections 2.1 and 2.2, and will be useful for us in Section 5. The next two lemmas (i.e., Lemmas 16 and 18) are variations that will be useful for us in Section 7.

Towards stating Lemma 15, let us recall the setting that was described in Sections 2.1 and 2.2: For a set $G \subseteq \{0,1\}^n$ of good objects, our goal is to find some $x \in G$; almost all objects are excellent, i.e. not only good but also in a subset $E \subseteq G$ with additional useful properties; there exists a distribution $\mathbf{T}$ over simple tests that distinguishes between excellent objects and objects that are not good; and the distribution $\mathbf{w}$ "fools" almost all tests $T \in \mathbf{T}$. In this case, $\mathbf{w}$ contains an object in G.

▶ **Lemma 15** (Randomized Tests). *Let $n \in \mathbb{N}$, and let $\epsilon_1, \epsilon_2, \epsilon_3, \epsilon_4, \epsilon_5 > 0$ be error parameters.*

- *Let $G \subseteq \{0,1\}^n$, and let $E \subseteq G$ such that $\Pr_{x \in \{0,1\}^n}[x \in E] \geq 1 - \epsilon_1$.*
- *Let $\boldsymbol{T}$ be a distribution over functions $T : \{0,1\}^n \to \{0,1\}$ such that for every $x \in E$ it holds that $\Pr_{T \sim \boldsymbol{T}}[T(x) = 1] \geq 1 - \epsilon_2$ and for every $x \notin G$ it holds that $\Pr_{T \sim \boldsymbol{T}}[T(x) = 0] \geq 1 - \epsilon_3$.*
- *Let $\boldsymbol{w}$ be a distribution that ϵ_5-fools all but an ϵ_4-fraction of the tests in $\boldsymbol{T}$; that is, the probability over $T \sim \boldsymbol{T}$ that $\left|\Pr[T(\boldsymbol{u}_n) = 1] - \Pr[T(\boldsymbol{w}) = 1]\right| > \epsilon_5$ is at most ϵ_4.*

Then, the probability that $\boldsymbol{w} \in G$ is at least $1 - (\epsilon_1 + \epsilon_2 + \epsilon_3 + 2\epsilon_4 + \epsilon_5)$.

Recall that in the proof of Theorem 3, the set of tests that are "fooled" by $\mathbf{w}$ is the set of tests that accept almost all of their inputs.

Proof of Lemma 15. Let $\mathcal{T}$ be the set of tests in the support of $\mathbf{T}$ that are ϵ_5-fooled by $\mathbf{w}$; that is, $\mathcal{T} = \left\{T \in \mathsf{supp}(\mathbf{T}) : \left|\Pr[T(\mathbf{u}_n) = 1] - \Pr[T(\mathbf{w}) = 1]\right| \leq \epsilon_5\right\}$. To upper-bound

the probability that $\mathbf{w} \notin G$, first note that a random test $T \sim \mathbf{T}$ accepts a random input $x \in \{0,1\}^n$ with high probability; this is the case because

$$\Pr_{T\sim\mathbf{T}}[T(\mathbf{u}_n) = 1] \geq \Pr[\mathbf{u}_n \in E] \cdot \min_{x\in E}\left\{\Pr_{T\sim\mathbf{T}}[T(x) = 1]\right\} \geq 1 - (\epsilon_1 + \epsilon_2) \ . \tag{4.1}$$

It follows that a random test $T \sim \mathbf{T}$ also accepts a *pseudorandom input* from the distribution $\mathbf{w}$ with high probability, since

$$\begin{aligned}\Pr_{T\sim\mathbf{T}}[T(\mathbf{w}) = 1] &\geq \Pr_{T\sim\mathbf{T}}[T \in \mathcal{T}] \cdot \Pr_{T\sim\mathbf{T}}[T(\mathbf{w}) = 1 | T \in \mathcal{T}] \\ &\geq (1-\epsilon_4) \cdot \left(\Pr_{T\sim\mathbf{T}}[T(\mathbf{u}_n) = 1 | T \in \mathcal{T}] - \epsilon_5\right) \\ &\geq (1-\epsilon_4) \cdot \left(\Pr_{T\sim\mathbf{T}}[T(\mathbf{u}_n) = 1] - \epsilon_4 - \epsilon_5\right) \ ,\end{aligned}$$

which, relying on Eq. (4.1), is lower-bounded by $1 - \epsilon_1 - \epsilon_2 - 2\epsilon_4 - \epsilon_5$.

However, if $\Pr[\mathbf{w} \notin G]$ is high, then there is significant probability that a random test from $\mathbf{T}$ will reject a pseudorandom input from $\mathbf{w}$. Specifically,

$$\Pr_{T\sim\mathbf{T}}[T(\mathbf{w}) = 0] \geq \Pr[\mathbf{w} \notin G] \cdot \min_{x\notin G}\left\{\Pr_{T\sim\mathbf{T}}[T(x) = 0]\right\} \geq \Pr[\mathbf{w} \notin G] - \epsilon_3 \ ,$$

and it follows that $\Pr[\mathbf{w} \notin G] \leq \epsilon_1 + \epsilon_2 + \epsilon_3 + 2\epsilon_4 + \epsilon_5$. ◀

We now present two variations on the argument above that are applicable in the setting of polynomials over finite fields. For a finite field $\mathbb{F}$, let $G \subseteq \mathbb{F}^n$, and assume that there exists a distribution $\mathbf{h}$ over polynomials $\mathbb{F}^n \to \mathbb{F}$ such that for every $x \notin G$ it holds that $\mathbf{h}(x) = 0$, with high probability. Further assume that there exists a hitting-set generator with high density for the polynomials h in the support of $\mathbf{h}$. Then, using an argument similar to the one in the beginning of Section 2.1, the hitting-set generator contains $x \in G$. [6]

▶ **Lemma 16** (Randomized Tests Over Finite Fields). *Let $n \in \mathbb{N}$, let $\mathbb{F}$ be any finite field, and let $\epsilon_1, \epsilon_2, \epsilon_3 > 0$ be three parameters. Assume that, for some $G \subseteq \mathbb{F}^n$, it holds that:*

1. *There exists a distribution $\boldsymbol{h}$ over $\{\mathbb{F}^n \to \mathbb{F}\}$ such that for every $x \notin G$ it holds that $\Pr_{h\sim\boldsymbol{h}}[h(x) = 0] \geq 1 - \epsilon_1$.*
2. *There exists a set $\mathcal{H} \subseteq \{\mathbb{F}^n \to \mathbb{F}\}$ such that $\Pr_{h\sim\boldsymbol{h}}[h \in \mathcal{H}] \geq 1 - \epsilon_2$.*
3. *There exists a distribution $\boldsymbol{w}$ over $\mathbb{F}^n$ such that for every $h \in \mathcal{H}$ it holds that $\Pr[h(\boldsymbol{w}) \neq 0] \geq 1 - \epsilon_3$.*

Then, $\Pr[\boldsymbol{w} \in G] \geq 1 - \epsilon_1 - \epsilon_2 - \epsilon_3$.

Proof. We first show that $\Pr[\mathbf{w} \in G] \geq \mathbb{E}_{h\sim\mathbf{h}}[\Pr[h(\mathbf{w}) \neq 0]] - \epsilon_1$. This is the case because

$$\begin{aligned}\mathbb{E}_{h\sim\mathbf{h}}\left[\Pr[h(\mathbf{w}) \neq 0]\right] &= \mathbb{E}_{x\sim\mathbf{w}}\left[\Pr_{h\sim\mathbf{h}}[h(x) \neq 0]\right] \\ &\leq \Pr_{x\sim\mathbf{w}}[x \in G] + \Pr_{x\sim\mathbf{w}}[x \notin G] \cdot \max_{x\notin G}\left\{\Pr_{h\sim\mathbf{h}}[h(x) \neq 0]\right\} \\ &\leq \Pr[\mathbf{w} \in G] + \epsilon_1 \ .\end{aligned}$$

[6] Recall that this argument is different than the argument in Lemma 15: On the one hand, we do not assume that G is dense, or that for every $x \in G$ it holds that $\mathbf{h}(x) \neq 0$, with high probability; but on the other hand, we require a hitting-set generator with high density for $h \in \mathtt{supp}(\mathbf{h})$ (rather than a pseudorandom generator).

Now, by our hypothesis, the probability that $\mathbf{h} \in \mathcal{H}$ is at least $1-\epsilon_2$, and for every $h \in \mathcal{H}$ it holds that $\Pr[h(\mathbf{w}) \neq 0] \geq 1-\epsilon_3$. Therefore,

$$\mathbb{E}_{h\sim\mathbf{h}}[\Pr[h(\mathbf{w}) \neq 0]] \geq \Pr_{h\sim\mathbf{h}}[h \in \mathcal{H}] \cdot \Pr_{h\sim\mathbf{h}}[h(\mathbf{w}) \neq 0 | h \in \mathcal{H}] \geq 1-\epsilon_2-\epsilon_3 \ ,$$

which implies that $\Pr[\mathbf{w} \in G] \geq 1-\epsilon_1-\epsilon_2-\epsilon_3$. ◀

In the next argument, instead of trying to hit a fixed set $G \subseteq \mathbb{F}^n$, we will fix a polynomial $p : \mathbb{F}^n \to \mathbb{F}$, and try to "fool" p (i.e., we want to construct a pseudorandom generator for p). Indeed, we will need to explain exactly what we mean by "fooling" in the context of functions over finite fields. Towards presenting the argument, let us first define the notion of *randomly computing p by a distribution of functions that is typically over simpler functions.*

▶ **Definition 17** (Randomly Computing a Function)**.** Let $\mathbb{F}$ be a finite field, let $p : \mathbb{F}^n \to \mathbb{F}$, and let $\mathcal{H}$ be a class of functions $\mathbb{F}^n \to \mathbb{F}$. For $\rho, \rho' > 0$, we say that p can be randomly computed with error ρ by a distribution $\mathbf{h}$ that is $(1-\rho')$-typically in $\mathcal{H}$, if:

1. For every $x \in \mathbb{F}^n$ it holds that $\Pr[p(x) = \mathbf{h}(x)] \geq 1-\rho$.
2. The probability that $\mathbf{h} \in \mathcal{H}$ is at least $1-\rho'$.

The following claim extends an argument that is implicit in the work of Bogdanov and Viola [3, Proof of Lemma 23]. Loosely speaking, our claim is the following: If p can be computed with small error by a distribution $\mathbf{h}$ that is typically in $\mathcal{H}$, then any distribution $\mathbf{w}$ over $\mathbb{F}^n$ that "fools" every $h \in \mathcal{H}$ also "fools" p, where "fooling" a function f means that for some (fixed) mapping $\xi : \mathbb{F} \to \mathbb{C}$ it holds that $\left|\mathbb{E}[\xi(f(\mathbf{w}))] - \mathbb{E}[\xi(f(\mathbf{u}_n))]\right|$ is small. [7]

▶ **Lemma 18** (An Extension of a Claim that is Implicit in [3])**.** *Let $n \in \mathbb{N}$, and let $\mathbb{F}$ be any finite field. Let $\epsilon_1, \epsilon_2, \epsilon_3 > 0$ be three parameters. Let $p : \mathbb{F}^n \to \mathbb{F}$, let $\mathcal{H} \subseteq \{\mathbb{F}^n \to \mathbb{F}\}$, and assume that p can be randomly computed with error ϵ_1 by a distribution $\boldsymbol{h}$ over $\{\mathbb{F}^n \to \mathbb{F}\}$ that is $(1-\epsilon_2)$-typically in $\mathcal{H}$.*

Let $\xi : \mathbb{F} \to \mathbb{C}$ be any mapping, and let $\delta = \max_{v,w\in\mathbb{F}}\{|\xi(v)-\xi(w)|\}$. Let $\boldsymbol{w}$ be a distribution over $\mathbb{F}^n$ such that for every $h \in \mathcal{H}$ it holds that $\left|\mathbb{E}[\xi(h(\boldsymbol{u}_n))] - \mathbb{E}[\xi(h(\boldsymbol{w}))]\right| < \epsilon_3$. Then, $\left|\mathbb{E}[\xi(p(\boldsymbol{u}_n))] - \mathbb{E}[\xi(p(\boldsymbol{w}))]\right| < 2\delta\cdot\epsilon_1 + \delta\cdot\epsilon_2 + \epsilon_3$.

Proof. For simplicity of notation, define $p' = \xi \circ p : \mathbb{F}^n \to \mathbb{C}$ and $h' = \xi \circ h : \mathbb{F}^n \to \mathbb{C}$. By the triangle inequality, we have that

$$\begin{aligned}\left|\mathbb{E}[p'(\mathbf{u}_n)] - \mathbb{E}[p'(\mathbf{w})]\right| \leq &\left|\mathbb{E}[p'(\mathbf{u}_n)] - \mathbb{E}_{h\sim\mathbf{h}}[h'(\mathbf{u}_n)]\right| + \\ &\left|\mathbb{E}_{h\sim\mathbf{h}}[h'(\mathbf{u}_n)] - \mathbb{E}_{h\sim\mathbf{h}}[h'(\mathbf{w})]\right| + \\ &\left|\mathbb{E}_{h\sim\mathbf{h}}[h'(\mathbf{w})] - \mathbb{E}[p'(\mathbf{w})]\right| . \end{aligned} \tag{4.2}$$

To upper bound the first term in Eq. (4.2), note that

$$\begin{aligned}\left|\mathbb{E}[p'(\mathbf{u}_n)] - \mathbb{E}_{h\sim\mathbf{h}}[h'(\mathbf{u}_n)]\right| &\leq \mathbb{E}_{u\in\mathbb{F}^n, h\sim\mathbf{h}}\left[\left|p'(u)-h'(u)\right|\right] \\ &\leq \mathbb{E}_{u\in\mathbb{F}^n}\left[\Pr_{h\sim\mathbf{h}}[h(u)\neq p(u)] \cdot \max_{v,w\in\mathbb{F}}\{|\xi(v)-\xi(w)|\}\right] \\ &\leq \delta\cdot\epsilon_1 \ ,\end{aligned}$$

[7] A standard choice for ξ is any fixed non-trivial character $\mathrm{e} : \mathbb{F} \to \mathbb{C}$.

where the last inequality holds because for every fixed $u \in \mathbb{F}^n$ it holds that $\Pr_{h\sim\mathbf{h}}[h(u) \neq p(u)] \leq \epsilon_1$. The third item is similarly upper bounded by $\delta \cdot \epsilon_1$, by replacing the uniform choice of $u \in \mathbb{F}^n$ with a choice of u according to the distribution $\mathbf{w}$.

To upper bound the second term in Eq. (4.2), note that

$$\begin{aligned}\left|\mathbb{E}_{h\sim\mathbf{h}}[h'(\mathbf{u}_n)] - \mathbb{E}_{h\sim\mathbf{h}}[h'(\mathbf{w})]\right| &\leq \mathbb{E}_{h\sim\mathbf{h}}\left[\left|\mathbb{E}[h'(\mathbf{u}_n)] - \mathbb{E}[h'(\mathbf{w})]\right|\right] \\ &\leq \Pr_{h\sim\mathbf{h}}[h \notin \mathcal{H}] \cdot \max_{v,w\in\mathbb{F}}\{|\xi(v) - \xi(w)|\} \\ &\quad + \mathbb{E}_{h\sim\mathbf{h}}\left[|\mathbb{E}[h'(\mathbf{u}_n)] - \mathbb{E}[h'(\mathbf{w})]|\middle| h \in \mathcal{H}\right] ,\end{aligned}$$

which is upper bounded by $\delta \cdot \epsilon_2 + \epsilon_3$. (Specifically, the first term is upper bounded by $\delta \cdot \epsilon_2$, whereas to bound the second term by ϵ_3 we use the hypothesis that for every $h \in \mathcal{H}$ it holds that $\left|\mathbb{E}[h'(\mathbf{u}_n)] - \mathbb{E}[h'(\mathbf{w})]\right| < \epsilon_3$.) ◀

5 Constant-depth circuits

5.1 Proof of Theorem 1

Let $c = D - d - 11$. Starting from a depth-d circuit $C : \{0,1\}^n \to \{0,1\}$, we will employ error-reduction within $\mathcal{AC}^0$, by first sampling inputs for C using the seeded extractor of Cheng and Li [5], and then taking the disjunction of the evaluation of C on these inputs. The extractor will be of depth $c + 10$, and will work for min-entropy $n'/\log^c(n')$, where n' is the number of random bits that it uses. Thus, this construction will yield a circuit $C' : \{0,1\}^{n'} \to \{0,1\}$ of depth $D = d + (c + 10) + 1$ that accepts all but $2^{n'/\log^c(n')} = 2^{n'/\log^{D-d-11}(n')}$ of its inputs. Details follow.

Let $C : \{0,1\}^n \to \{0,1\}$ be a circuit of depth d. We will rely on the following theorem from [5], which we cite with minor changes of notation:

▶ **Theorem 19** (An $\mathcal{AC}^0$-Computable Seeded Extractor [5, Thm 1.5]). *For any constant $c \in \mathbb{N}$, and $k = \Omega(n'/\log^c(n'))$ and any $\epsilon = 1/\mathrm{poly}(n')$, there exists an explicit construction of a strong (k,ϵ)-extractor $Ext : \{0,1\}^{n'} \times \{0,1\}^d \to \{0,1\}^n$ that can be computed by an $\mathcal{AC}^0$ circuit of depth $c + 10$, where $d = O(\log(n))$, $n = k^{\Omega(1)}$ and the extractor family has locality $O(\log^{c+5}(n))$.*

We will not need the strongness property or the locality property in the current proof. Let $n' = \mathrm{poly}(n)$ such that for $k = \Omega(n'/\log^c(n'))$ it holds that $n = k^{\Omega(1)}$, and let $Ext : \{0,1\}^{n'} \times \{0,1\}^d \to \{0,1\}^n$ be the seeded extractor from Theorem 19, instantiated with error parameter $\epsilon = 1/4$. We construct a circuit $C' : \{0,1\}^{n'} \to \{0,1\}$ that first computes the values $Ext(x,z)$, for each possible seed $z \in \{0,1\}^d$, then evaluates C on each value $E(x,z)$, and finally takes an OR of these evaluations; that is, $C'(x) = \vee_{z\in\{0,1\}^s} C\left(Ext(x,z)\right)$.

Note that C' has depth D and size $\mathrm{poly}(n)$. Also note that the number of inputs $x \in \{0,1\}^{n'}$ for which $\Pr_z[C(Ext(x,z))] < 1/4$ is at most $2^{n'/\log^c(n')}$. [8] In particular, C' accepts all but at most $2^{n'/\log^c(n')}$ of its inputs, and for each satisfying input x for C', we can find a corresponding satisfying input for C among $\{Ext(x,z)\}_{z\in\{0,1\}^s}$.

[8] Otherwise, the uniform distribution on such inputs yields a source X of min-entropy $n'/\log^c(n')$ such that C distinguishes $Ext(X)$ from the uniform distribution over $\{0,1\}^n$ with probability $1/4$.

5.2 Proofs of Theorems 2 and 3

The first step towards proving Theorems 2 and 3 is to establish a derandomized switching lemma that simplifies depth-2 formulas of *bounded-width*; after presenting several required definitions in Section 5.2.1, we prove the lemma in Section 5.2.2. Then, in Section 5.2.3, we use the lemma to prove Theorems 2 and 3.

5.2.1 Preliminary definitions, and results from [10]

For any restriction $\rho \in \{0,1,\star\}^n$, denote by $\mathfrak{C}(\rho)$ the subcube that corresponds to the living variables under ρ; that is, $\mathfrak{C}(\rho) = \{x \in \{0,1\}^n : \forall i \in [n] \text{ s.t. } \rho_i \neq \star \text{ it holds that } x_i = \rho_i\}$. We identify strings $r \in \{0,1\}^{(q+1)\cdot n}$, where $n, q \in \mathbb{N}$, with restrictions $\rho = \rho_r \in \{0,1,\star\}^n$, as follows: Each variable is assigned a block of $q+1$ bits in the string; the variable remains alive if the first q bits in the block are all zeroes, and otherwise takes the value of the $(q+1)^{th}$ bit. When we refer to a "block" in the string that corresponds to a restriction, we mean a block of $q+1$ bits that corresponds to some variable. When we say that a restriction is chosen from a distribution $\mathbf{r}$ over $\{0,1\}^{(q+1)\cdot n}$, we mean that a string is chosen according to $\mathbf{r}$, and interpreted as a restriction. Moreover, when we say that an algorithm "reads bits" in the restriction, we mean that it reads bits in the corresponding string.

In addition, we will sometimes identify a *pair* of strings $y \in \{0,1\}^{q\cdot n}$ and $z \in \{0,1\}^n$ with a restriction $\rho = \rho_{y,z}$. In this case, the restriction $\rho = \rho_{y,z}$ is the restriction ρ_r that is obtained by combining y and z to a string r in the natural way (i.e., appending a bit from z to each block of q bits in y). Note that the string y determines which variables ρ keeps alive, and the string z determines the values that ρ assigns to the fixed variables.

Throughout the section, whenever we consider a depth-2 formula for a function $F : \{0,1\}^n \to \{0,1\}$, we allow the formula to be a redundant representation of F (i.e., not necessarily the most concise representation of F as a formula), and in particular we allow formulas in which some clauses are simply constants. We will identify any clause of a depth-2 formula with the corresponding subset of the literals; the clause is a conjunction of the literals if the formula is a DNF, and otherwise it is a disjunction of the literals. We say that a function $F^{\mathtt{low}} : \{0,1\}^n \to \{0,1\}$ is lower-sandwiching for F if for every $x \in \{0,1\}^n$ it holds that $F^{\mathtt{low}}(x) \leq F(x)$. Similarly, we say that $F^{\mathtt{up}} : \{0,1\}^n \to \{0,1\}$ is upper-sandwiching for F if for every $x \in \{0,1\}^n$ it holds that $F(x) \leq F^{\mathtt{up}}(x)$.

5.2.1.1 Refinements: Definition and basic facts

We need several definitions that are related to the results of Gopalan, Meka, and Reingold [10]. Their main theorem involves a process of *sparsification* of a depth-2 formula. The sparsification process is iterative: In each iteration, they identify a *quasi-sunflower* in the formula (a notion that was introduced by Rossman [18]), and simplify the quasi-sunflower using one of two operations. The first operation is simply the removal of a clause from the formula; and the second operation is the removal of a set $f_1, ..., f_u$ of $u \geq 2$ clauses, replacing them with a new clause that consists of the set of literals that are shared by all the u clauses (i.e., replacing $f_1, ..., f_u$ with the clause $\bigcap_{j\in[u]} f_j$). The following definition generalizes this sparsification process. [9]

[9] The reason that we need this generalization is in order to facilitate the proof of Claim 23; this is also the reason that we allow formulas to have redundant clauses that compute constant functions.

▶ **Definition 20** (Refinements of a Depth-2 Formula)**.** Let $F : \{0,1\}^n \to \{0,1\}$ be a depth-2 formula with at least two clauses. We define the following three syntactic operations on F, which we call refinement steps.

1. A removal step is simply the removal of a clause from F.
2. A merging step is the removal of $u \geq 2$ clauses $f_1, ..., f_u$ from F, and the addition of a new clause that consists of the set of literals that appear in all the u clauses (i.e., replacing $f_1, ..., f_u$ with the new clause $\bigcap_{j\in[u]} f_j$). If $\bigcap_{j\in[u]} f_j = \emptyset$, then the new clause computes the constant one function if F is a DNF, and the constant zero function if F is a CNF.
3. A clean-up step is the removal of one or more clauses that compute the constant zero function from a DNF, or of one or more clauses that compute the constant one function from a CNF.

We say that a depth-2 formula $F' : \{0,1\}^n \to \{0,1\}$ is a refinement of another depth-2 formula $F : \{0,1\}^n \to \{0,1\}$ if F' can be obtained from F either by a sequence of removal steps and clean-up steps, or by a sequence of merging steps and clean-up steps.

We now state some basic facts about refinements, which will be useful for us later on. The following two facts follow from Definition 20:

▶ **Fact 21** (Refinements Under Negations)**.** *Let $F : \{0,1\}^n \to \{0,1\}$ and $F' : \{0,1\}^n \to \{0,1\}$ be depth-2 formulas. Then, F' is a refinement of F if and only if $\neg(F')$ is a refinement of $\neg F$.*

▶ **Fact 22** (Sandwiching Refinements)**.** *Let $F : \{0,1\}^n \to \{0,1\}$ be a DNF. Then, any refinement of F that is obtained by a sequence of removal steps and clean-up steps is lower-sandwiching for F, and any refinement of F that is obtained by a sequence of merging steps clean-up steps is upper-sandwiching for F.*

Loosely speaking, the following claim asserts that if F' is a refinement of F, then for any restriction ρ it holds that $(F')\restriction_\rho$ is a refinement of $F\restriction_\rho$. That is, intuitively, restricting both F and F' by ρ does not affect the fact that the latter formula is a refinement of the former.

▶ **Claim 23** (Refinements Under Restrictions)**.** *Let $F : \{0,1\}^n \to \{0,1\}$ be a depth-2 formula of width w and size m, and let $F' : \{0,1\}^n \to \{0,1\}$ be a refinement of F. Then, for any restriction $\rho \in \{0,1,\star\}^n$ it holds that $F\restriction_\rho$ can be computed by a depth-2 formula Φ of width w and size m such that $F'\restriction_\rho$ is a refinement of Φ.*

The proof of Claim 23 relies on an elementary (and tedious) case analysis, so we defer it to Appendix B.

5.2.1.2 Two theorems from [10]

For $\epsilon > 0$ and two Boolean functions F and F' over a domain $\mathfrak{D}$, we say that F and F' are ϵ-close if $\Pr_{x\in\mathfrak{D}}[F(x) = F'(x)] \geq 1 - \epsilon$. We say that F' is an ϵ-refinement of F if F' is both a refinement of F, and ϵ-close to F. Similarly, we say that F' is an ϵ-lower-sandwiching refinement (resp., ϵ-upper-sandwiching refinement) of F if F' is both ϵ-close to F and a lower-sandwiching (resp., upper-sandwiching) refinement of F. Then, the main result of Gopalan, Meka, and Reingold [10] can be stated as follows:

▶ **Theorem 24** ([10, Thm 1.2])**.** *Let $F : \{0,1\}^n \to \{0,1\}$ be a depth-2 formula of width w, and let $\beta > 0$. Then, there exist β-lower-sandwiching and β-upper-sandwiching refinements of F, denoted by $F^{\texttt{low}}$ and $F^{\texttt{up}}$, respectively, such that the size of $F^{\texttt{low}}$ and of $F^{\texttt{up}}$ is at most $m' = 2^{\tilde{O}(w)\cdot\log\log(1/\beta)}$, and their width is at most w.*

We will also need a pseudorandom generator construction from [10]. In fact, we will rely on an assertion from the proof of their generator construction.

▶ **Theorem 25** ([10, In the proof of Thm 3.1]). *Let $F : \{0,1\}^n \to \{0,1\}$ be a depth-2 formula of width w, and let $\delta_0 > 0$. Then, every δ-almost t-wise independent distribution δ_0-fools F, where $\log(1/\delta) = O(w^2 \cdot \log^2(w) + w \cdot \log(w) \cdot \log(1/\delta_0))$ and $t = O(w^2 \cdot \log(w) + w \cdot \log(1/\delta_0))$.*

5.2.2 Width-dependent derandomizations of the switching lemma

In the proposition statements in this section, the letter n denotes the number of input bits for a formula, the number of clauses (i.e., size) is denoted by m, the width is denoted by w, and $\delta > 0$ is an error parameter (which will typically take the value $\delta = 1/\text{poly}(n)$ in our applications). As a first step, we need to adapt the derandomized switching lemma of Trevisan and Xue [23] such that it will depend on the width of the depth-2 formula that we wish to "switch". Then, we will state and prove our new derandomized switching lemma, which is the main technical part in this section.

▶ **Proposition 26** (An Adaptation of the Derandomized Switching Lemma of [23]). *Let $m : \mathbb{N} \to \mathbb{N}$, let $w : \mathbb{N} \to \mathbb{N}$ such that $w(n) \leq O(\log(m(n)))$, and let $\delta : \mathbb{N} \to [0,1)$ such that $\delta(n) \leq 2^{-O(w(n))}$. Let $\boldsymbol{r}$ be a distribution over $\{0,1\}^{O(\log(w)) \cdot n}$ that is δ'-almost t'-wise independent, where $\log(1/\delta') = O(t') = \tilde{O}(w) \cdot \log(1/\delta) \cdot \log(m) + O(\log(n/\delta))$. Then, for any depth-2 formula $F : \{0,1\}^n \to \{0,1\}$ of width $w = w(n)$ and size $m = m(n)$, with probability at least $1 - 2\delta$ (where $\delta = \delta(n)$) over choice of $\rho \sim \boldsymbol{r}$ it holds that:*
1. *The restricted formula $F\restriction_\rho$ can be computed by a decision tree of depth $D = O(\log(1/\delta))$.*
2. *The number of variables that are kept alive by ρ is at least $\Omega(n/w)$.*

In particular, a restriction $\rho \sim \boldsymbol{r}$ can be sampled using a seed of length $\tilde{O}(w) \cdot \log(1/\delta) \cdot \log(m) + O(\log(n/\delta))$.

Proof. Loosely speaking, the main lemma of Trevisan and Xue [23] reduces the task of finding a restriction that simplifies F to the task of "fooling" a large number of auxiliary CNFs. Going through their proof, we observe is that if F has width w, then each of the auxiliary CNFs also has width (roughly) w; that is, their proof can be adapted to show the following:

▶ **Lemma 27** (A Variation on [23, Lemma 7]). *Let $F : \{0,1\}^n \to \{0,1\}$ be a depth-2 formula of size m and width w. For $q \in \mathbb{N}$ and $p = 2^{-q}$, let $\rho \in \{0,1,\star\}^n$ be a restriction that is chosen according to a distribution over $\{0,1\}^{(q+1)\cdot n}$ that δ_0-fools all CNFs of width $w' = w \cdot (q+1)$. Then, the probability that $F\restriction_\rho$ cannot be computed by a decision tree of depth D is at most $2^{D+w+1} \cdot (5pw)^D + \delta_0 \cdot 2^{(D+1)\cdot(2\cdot w + \log(m))}$.*

The proof of Lemma 27 is a relatively straightforward adaptation of the original proof in [23], so we defer it to Appendix B. We will use the lemma with the parameters $p = 1/O(w)$ and $\delta_0 = 2^{-O(D\cdot(w+\log(m)))}$, in order to get the probability of error down to δ. Relying on Theorem 25, the auxiliary CNFs of width w' are δ_0-fooled by $\mathbf{r}$, [10] and therefore with probability $1 - \delta$ it holds that $F\restriction_\rho$ can be computed by a decision tree of depth D.

The expected number of variables that the pseudorandom restriction leaves alive is $\Omega(n/w)$ (because the distribution on each block of $O(\log(w))$ bits in $\mathbf{r}$, which corresponds

[10] This is because according to Theorem 25, CNFs of width w' are δ_0-fooled by any distribution that is δ''-almost t''-wise independent, where $t'' = O\left((w')^2 \cdot \log(w') + w' \cdot \log(1/\delta_0)\right) = \tilde{O}(w) \cdot \log(1/\delta) \cdot \log(m)$ and $\log(1/\delta'') = O\left((w')^2 \cdot \log^2(w') + w' \cdot \log(w') \cdot \log(1/\delta_0)\right) = \tilde{O}(w) \cdot \log(1/\delta) \cdot \log(m)$.

to a variable, is of statistical distance at most δ' from uniform, where $\delta' < 2^{-w}$). Since $\mathbf{r}$ is δ'-almost t'-wise independent, where $\delta' < 1/\mathrm{poly}(n/\delta)$ and $t' > O(\log(w))$, the blocks in $\mathbf{r}$ that correspond to each variable are $\frac{1}{\mathrm{poly}(n/\delta)}$-almost $O(1)$-wise independent. Relying on Fact 13, the probability that $\Omega(n/w)$ variables remain alive is at least $1-\delta$. ◀

We mention that the derandomized switching lemma of Goldreich and Wigderson [9, second step of the proof of Lemma 3.3] can also be adapted to depend on the width w of the formula that we want to "switch"; in this case, the required seed length is $\tilde{O}(w) \cdot 2^w \cdot \log(1/\delta)$, where δ is the probability of error (and the target depth of the decision tree is $D = O(\log(1/\delta))$). We provide the details in Appendix B. We now turn to state the new width-dependent derandomization of the switching lemma and prove it:

▶ **Proposition 28** (A New Width-Dependent Derandomization of the Switching Lemma)**.** *Let $m : \mathbb{N} \to \mathbb{N}$, let $w : \mathbb{N} \to \mathbb{N}$ such that $w(n) \leq O(\log(m(n)))$, let $\delta : \mathbb{N} \to [0,1)$, and let $\alpha : \mathbb{N} \to [0,1)$. Let $\delta' > 0$ and $t' \in \mathbb{N}$ such that $\log(1/\delta') = O(t') = \tilde{O}(w^2) \cdot \log(1/\delta) \cdot \log\log(m/\alpha\delta) + \tilde{O}(w) \cdot \log(m/\alpha\delta) + O(\log(n/\delta))$. Let $\boldsymbol{y}$ be a distribution over $\{0,1\}^{O(\log(w))\cdot n}$ that is δ'-almost t'-wise independent, and let $\boldsymbol{z}$ be a distribution over $\{0,1\}^n$ that is δ'-almost t'-wise independent. Finally, let $\rho = \rho_{\boldsymbol{y},\boldsymbol{z}}$ be a restriction that is chosen by using a sample from $\boldsymbol{y}$ to determine which variables are kept alive, and an independent sample from $\boldsymbol{z}$ to determine values for the fixed variables.*

Then, for any depth-2 formula $F : \{0,1\}^n \to \{0,1\}$ of width $w = w(n)$ with $m = m(n)$ clauses, with probability at least $1 - 4\delta$ (where $\delta = \delta(n)$) over choice of ρ it holds that:

1. *There exists a lower-sandwiching refinement $F^{\mathtt{low}}$ of F such that $F^{\mathtt{low}}{\restriction_\rho}$ and $F{\restriction_\rho}$ are α-close (i.e., $\Pr_{x \in \mathfrak{C}(\rho)}[F^{\mathtt{low}}(x) = F(x)] \geq 1-\alpha$) and such that the restricted refinement $F^{\mathtt{low}}{\restriction_\rho}$ can be computed by a decision tree of depth $D = O(\log(1/\delta))$.*
2. *There exists an upper-sandwiching refinement $F^{\mathtt{up}}$ of F such that $F^{\mathtt{up}}{\restriction_\rho}$ and $F{\restriction_\rho}$ are α-close and such that $F^{\mathtt{up}}{\restriction_\rho}$ can be computed by a decision tree of depth $D = O(\log(1/\delta))$.*
3. *The number of variables that are kept alive by ρ is at least $\Omega(n/w)$.*

In particular, a restriction ρ can be sampled using a seed of length $\tilde{O}(w^2) \cdot \log(1/\delta) \cdot \log\log(m/\alpha\delta) + \tilde{O}(w) \cdot \log(m/\alpha\delta) + O(\log(n/\delta))$.

Note that when $m = \Theta(1/\delta) = \Theta(1/\alpha) = \mathrm{poly}(n)$, the seed length in Proposition 28 is $\tilde{O}(w^2 \cdot \log(n))$. As in the overview in Section 2.2, our strategy in the proof of Proposition 28 will be as follows. Let $F^{\mathtt{low}}$ and $F^{\mathtt{up}}$ be the refinements of F from Theorem 24. Using the fact that $F^{\mathtt{low}}$ and $F^{\mathtt{up}}$ are of width w and of size $2^{\tilde{O}(w)\cdot\log\log(m/\alpha\delta)}$, we will rely on Proposition 26 to prove that, with high probability, both $F^{\mathtt{low}}{\restriction_\rho}$ and $F^{\mathtt{up}}{\restriction_\rho}$ simplify to depth-D decision trees. The main challenge will be to prove that with high probability it holds that $F^{\mathtt{low}}{\restriction_\rho}$ (resp., $F^{\mathtt{up}}{\restriction_\rho}$) and $F{\restriction_\rho}$ are α-close. The following lemma is the key one needed to establish the latter assertion, and after proving the lemma, we will use it to prove Proposition 28.

▶ **Lemma 29.** *Let $m : \mathbb{N} \to \mathbb{N}$, let $w : \mathbb{N} \to \mathbb{N}$ such that $w(n) \leq O(\log(m(n)))$, and let $\delta : \mathbb{N} \to [0,1)$. Let $F : \{0,1\}^n \to \{0,1\}$ be a depth-2 formula of size $m = m(n)$ and width $w = w(n)$. For $\alpha > 0$ and $\beta \leq \frac{\alpha^6 \cdot (\delta/4)^4}{m^4 \cdot \log^6(1/\delta)}$, let $F' : \{0,1\}^n \to \{0,1\}$ be a β-refinement of F.*

Fix $I \subseteq [n]$, and let $\boldsymbol{z}$ be a distribution over $\{0,1\}^n$ that β-fools all DNFs of width w. Let $\rho = \rho_{I,\boldsymbol{z}} \in \{0,1,\star\}^n$ be the restriction that is obtained by fixing values to the variables indexed by $[n] \setminus I$ according to the corresponding bits of $\boldsymbol{z}$. Then, with probability at least $1-\delta$ over choice of $\boldsymbol{z}$ it holds that $F'{\restriction_\rho}$ is an α-refinement of a depth-2 formula of size m and width w for $F{\restriction_\rho}$.

Proof. We will prove the claim assuming that F is a DNF; if F is a CNF, then we can rely on Fact 21 to deduce that the assertion of the lemma holds for F if and only if it holds

for the DNF $\neg F$. Also note that by Claim 23, for any $\rho \in \{0,1,\star\}^n$ it holds that $F'|_\rho$ is a refinement of a depth-2 formula of size m and width w for $F|_\rho$. Thus, we only need to prove that with probability at least $1-\delta$ it holds that $F'|_\rho$ is α-close to $F|_\rho$. Recall that $I \subseteq [n]$ is fixed throughout the proof; for brevity of notation, for any $z \in \{0,1\}^n$ denote $\rho_z = \rho_{I,z}$.

In high-level, the proof follows the overview that was presented in Section 2.2, and in particular relies on Lemma 15. We first define a set E of excellent restrictions, which are restrictions ρ such that $F'|_\rho$ is $\sqrt{\beta}$-close to $F|_\rho$, and show that almost all restrictions are excellent. We will then define a set B of bad restrictions, which are restrictions ρ such that $F'|_\rho$ is not α-close to $F|_\rho$. After defining E and B we will define the distribution $\mathbf{T}$ over tests that accepts, with high probability, every restriction in E, and rejects, with high probability, every restriction in B. Then, we will show that the residual tests $T \in \mathbf{T}$ are relatively "simple", in the sense that they can be computed by depth-3 circuits with a specific structure (i.e., top AND gate and bottom fan-in w). And finally, we will show a hitting-set generator for the set of tests in the support of $\mathbf{T}$ that accept almost all of their input restrictions, and conclude the argument using Lemma 15.

Excellent restrictions and bad restrictions. For any $\rho \in \{0,1,\star\}^n$, let $\mathtt{err}(\rho) = \Pr_{x\in\mathfrak{C}(\rho)}[F'(x) \neq F(x)]$ be the fraction of inputs in $\mathfrak{C}(\rho)$ on which F and F' disagree. Our goal is to show that $\Pr_{z\sim\mathbf{z}}[\mathtt{err}(\rho_z) \leq \alpha] \geq 1-\delta$. Consider the following two sets:

▶ **Definition 29.1** (Excellent and Bad Restrictions). *Let $E = \{z \in \{0,1\}^n : \mathtt{err}(\rho_z) \leq \sqrt{\beta}\}$ be the set of* excellent *choices of restrictions, and let $B = \{z \in \{0,1\}^n : \mathtt{err}(\rho_z) > \alpha\}$ be the set of* bad *choices of restrictions.*

Since F' is β-close to F, a random restriction $\rho_{I,\mathbf{u}_n}$ is excellent with probability at least $1-\sqrt{\beta}$. [11] We want to show that a pseudorandom restriction $\rho_{\mathbf{z}} = \rho_{I,\mathbf{z}}$ is not bad, with probability at least $1-\delta$.

A distribution over simple tests. Let $t = O(\log(1/\delta)/\alpha)$. We now define a distribution $\mathbf{T}$ over tests $\{0,1\}^n \to \{0,1\}$, such that the random variable $\mathbf{T}(z)$ will essentially be the result of the following random test: Given $z \in \{0,1\}^n$, the test uniformly samples t inputs in $\mathfrak{C}(\rho_z)$, and accepts z if and only if F and F' agree on all the t inputs.

For $x \in \{0,1\}^{|I|}$ and $z \in \{0,1\}^n$, denote by $x|_z \in \mathfrak{C}(\rho_z)$ the string that is obtained by fixing the variables indexed by I according to x, and the rest of the variables (i.e., the ones indexed by $[n] \setminus I$) according to the corresponding bits from z. For any $x \in \{0,1\}^{|I|}$, let $T_x : \{0,1\}^n \to \{0,1\}$ be the function such that $T_x(z) = 1$ if and only if $F'(x|_z) = F(x|_z)$. Also, for $\bar{x} = x^{(1)}, ..., x^{(t)} \in \{0,1\}^{t\cdot|I|}$, let $T_{\bar{x}}$ be the function $T_{\bar{x}}(z) = \wedge_{i=1}^t T_{x^{(i)}}(z)$. Finally, let $\mathbf{T}$ be the distribution over tests that is obtained by uniformly choosing $\bar{x} \in \{0,1\}^{t\cdot|I|}$ and outputing $T_{\bar{x}}$. Note that $\mathbf{T}(z)$ is indeed the result of uniformly sampling t inputs in $\mathfrak{C}(\rho_z)$, and accepting z if and only if F' and F agree on all the t sampled inputs.

By our choice of the parameter t, and since β is sufficiently small, the distribution $\mathbf{T}$ indeed distinguishes between E and B:

▶ **Fact 29.2.** *For any $z \in E$ it holds that $\Pr_{T\sim\mathbf{T}}[T(z)=1] \geq (1-\sqrt{\beta})^t \geq 1 - t\cdot\sqrt{\beta}$, and for any $z \in B$ it holds that $\Pr_{T\sim\mathbf{T}}[T(z)=1] < (1-\alpha)^t < \delta/3$.*

For $\eta = \sqrt{t+1}\cdot\beta^{1/4}$, let $\mathbf{T}'$ be the set of tests $T_{\bar{x}} \in \mathbf{T}$ that accept at least $1-\eta$ of their inputs (i.e., $\mathbf{T}' = \{T_{\bar{x}} : \Pr_{z\in\{0,1\}^n}[T_{\bar{x}}(z)=1] \geq 1-\eta\}$). We will abuse the notations $\mathbf{T}$

[11] Because $\mathbb{E}[\mathtt{err}(\rho_{I,\mathbf{u}_n})] = \Pr_{x\in\{0,1\}^n}[F'(x) \neq F(x)] \leq \beta$, which implies that $\Pr[\mathtt{err}(\rho_{I,\mathbf{u}_n}) > \sqrt{\beta}] < \sqrt{\beta}$.

and $\mathbf{T}'$, by using them both to denote sets and to denote the uniform distribution over the corresponding set. To see that the set $\mathbf{T}'$ is dense in $\mathbf{T}$, note that

$$\mathbb{E}_{T_{\bar{x}}\in\mathbf{T}}\left[\Pr_{z\in\{0,1\}^n}[T_{\bar{x}}(z)=1]\right] = \mathbb{E}_{z\in\{0,1\}^n}\left[\Pr_{T_{\bar{x}}\in\mathbf{T}}[T_{\bar{x}}(z)=1]\right]$$
$$\geq \Pr_{z\in\{0,1\}^n}[z\in E]\cdot\min_{z\in E}\left\{\Pr_{T_{\bar{x}}\in\mathbf{T}}[T_{\bar{x}}(z)=1]\right\},$$

which is at least $1-\sqrt{\beta}-t\cdot\sqrt{\beta}=1-\eta^2$. Therefore, the probability over $T_{\bar{x}}\in\mathbf{T}$ that $T_{\bar{x}}$ rejects more than η of its input restrictions is at most η.

A hitting-set generator for $\mathbf{T}'$. Towards designing a hitting-set generator with high density for every $T_{\bar{x}}\in\mathbf{T}'$, we first show that each $T_{\bar{x}}\in\mathbf{T}$ can be computed by a depth-3 circuit with a top AND gate and small bottom fan-in. To do so, we first show that for a single $x\in\{0,1\}^{|I|}$ (rather than for $\bar{x}=x^{(1)},...,x^{(t)}$) it holds that T_x can be computed by a depth-3 circuit with a top AND gate and small bottom fan-in.

▶ **Claim 29.3.** *For every fixed $x\in\{0,1\}^{|I|}$, the function $T_x:\{0,1\}^n\to\{0,1\}$ can be computed by a depth-3 circuit with a top AND gate of fan-in at most m such that the bottom fan-in of the circuit is at most w.*

Proof. Denote the number of refinement steps that were applied to F to obtain F' by $k\leq m$. For any $i\in[k]$, let $F^{(i)}$ be the formula in the beginning of the i^{th} refinement step in the transformation of F to F', and let $F^{(k+1)}=F'$. Note that $T_x(z)=1$ if and only if for every $i\in[k]$ it holds that $F^{(i)}(x{\restriction}_z)=F^{(i+1)}(x{\restriction}_z)$ (one direction is immediate, whereas the other direction follows by the monotonicity of the sequence $F^{(1)}(x{\restriction}_z),...,F^{(k+1)}(x{\restriction}_z)$ [12]).

For every $i\in[k]$, let $T_{x,i}$ be the function such that $T_{x,i}(z)=1$ if and only if $F^{(i)}(x{\restriction}_z)=F^{(i+1)}(x{\restriction}_z)$. We will show that each $T_{x,i}$ can be computed by a DNF of width w. This claim suffices to conclude the proof, since it implies that T_x can be computed by a circuit with a top AND gate that is connected to $k\leq m$ DNFs of width w. To prove the claim, fix $i\in[k]$, and let us conduct a case analysis:

- If the i^{th} refinement step was a clean-up step, then $T_{x,i}\equiv 1$.
- If the i^{th} step was a removal step, then let $f^{(i)}$ be the clause that was removed from $F^{(i)}$ in the i^{th} step, and let $F^{(i+1)}=\left(F^{(i)}\setminus f^{(i)}\right)$ be the formula that is obtained by dropping the clause $f^{(i)}$ from $F^{(i)}$. Note that $F^{(i+1)}(x{\restriction}_z)=F^{(i)}(x{\restriction}_z)$ if and only if either $f^{(i)}(x{\restriction}_z)=0$ or $\left(F^{(i)}\setminus f^{(i)}\right)(x{\restriction}_z)=1$. The latter event is a disjunction of at most m events (because $\left(F^{(i)}\setminus f^{(i)}\right)$ is a DNF of size at most $m-1$), each of which depends on the values of at most w bits in $x{\restriction}_z$. Thus, each of the (at most m) events depends on at most w bits in z, and can therefore be decided by a DNF of width w. It follows that $T_{x,i}$ is the disjunction of width-w DNFs, which is a width-w DNF.
- If the i^{th} refinement step in the transformation of F to F' was a merging step, denote the $u\geq 2$ clauses that were removed from $F^{(i)}$ in the step by $f_1^{(i)},...,f_u^{(i)}$, and the new clause that was added in their stead by $h^{(i)}$. Note that $F^{(i+1)}(x{\restriction}_z)=F^{(i)}(x{\restriction}_z)$ if and only if either $h^{(i)}(x{\restriction}_z)=0$ or $F^{(i)}(x{\restriction}_z)=1$. This is a disjunction of at most $m+1$ events, each of which depends on at most w bits in $x{\restriction}_z$ (and thus on at most w bits in z). Thus, in this case too it holds that $T_{x,i}$ can be computed by a DNF of width w. ◀

[12] If F' was obtained by merging steps and clean-up steps, then $F^{(1)}(x{\restriction}_z)\leq...\leq F^{(k+1)}(x{\restriction}_z)$, whereas if F' was obtained by removal steps and clean-up steps, then $F^{(1)}(x{\restriction}_z)\geq...\geq F^{(k+1)}(x{\restriction}_z)$.

For a fixed $\bar{x} = x^{(1)}, ..., x^{(t)} \in \{0,1\}^{t \cdot |I|}$, we can compute $T_{\bar{x}}$ by taking a conjunction of t circuits for the corresponding T_x's (i.e., $\wedge_{i \in [t]} T_{x^{(i)}}$), which is a depth-3 circuit with bottom fan-in at most w and top fan-in at most $t \cdot m$. We are now ready to prove that $\mathbf{z}$ is a hitting-set generator with density $1 - \delta/3$ for every $T_{\bar{x}} \in \mathbf{T}'$:

▶ **Claim 29.4.** *For every $T_{\bar{x}} \in \boldsymbol{T}'$ it holds that* $\Pr[T(\boldsymbol{z}) = 1] \geq 1 - \delta/3$.

Proof. Fix $T_{\bar{x}} \in \mathbf{T}'$, and recall that by the definition of $\mathbf{T}'$ it holds that $T_{\bar{x}}$ accepts at least $1 - \eta$ of its inputs. Thus, each of the DNFs in the middle layer of the circuit that we constructed for $T_{\bar{x}}$ accepts $1 - \eta$ of the inputs. It follows that when using the distribution $\mathbf{z}$, which is β-pseudorandom for such DNFs, each of these DNFs accepts with probability at least $1 - \eta - \beta$. By a union-bound, it follows that

$$\begin{aligned}\Pr_{z \sim \mathbf{z}}[T_{\bar{x}}(z) = 1] &\geq 1 - (\eta + \beta) \cdot (t \cdot m) \\ &> 1 - (2 \cdot t \cdot m) \cdot \eta \\ &= 1 - O\left((\log(1/\delta)/\alpha)^{3/2} \cdot m \cdot \beta^{1/4}\right) ,\end{aligned}$$

which is larger than $1 - \delta/3$ by the hypothesis that β is sufficiently small. ◀

Invoking Lemma 15. We now conclude the argument by invoking Lemma 15. Let E be as in Definition 29.1, and let $G = \{0,1\}^n \setminus B$; recall that for $\epsilon_1 = \sqrt{\beta}$ it holds that $\Pr_{z \in \{0,1\}^n}[z \in E] \geq 1 - \epsilon_1$. Denoting $\epsilon_2 = t \cdot \sqrt{\beta}$ and $\epsilon_3 = \delta/3$, according to Fact 29.2, for any $z \in E$ it holds that $\Pr_{T_{\bar{x}} \sim \mathbf{T}}[T_{\bar{x}}(z) = 1] \geq 1 - \epsilon_2$ and for any $z \notin G$ it holds that $\Pr_{T_{\bar{x}} \sim \mathbf{T}}[T_{\bar{x}}(z) = 0] \geq 1 - \epsilon_3$.

Finally, for $\epsilon_4 = \eta$ it holds that the set $\mathbf{T}'$ is of density at least $1 - \epsilon_4$ in $\mathbf{T}$, and for every $T_{\bar{x}} \in \mathbf{T}'$, by Claim 29.4 it holds that $\mathbf{z}$ fools $T_{\bar{x}}$ with error at most $\epsilon_5 = \delta/3$ (because $\Pr_{z \in \{0,1\}^n}[T_{\bar{x}}(z) = 1] \geq 1 - \eta \geq 1 - \delta/3$ and $\Pr_{z \sim \mathbf{z}}[T_{\bar{x}}(z) = 1] \geq 1 - \delta/3$). Relying on Lemma 15, the probability that $\mathbf{z} \notin G$ is at most

$$\sqrt{\beta} + t \cdot \sqrt{\beta} + \delta/3 + 2 \cdot \eta + \delta/3 = 2\delta/3 + \eta^2 + 2 \cdot \eta < \delta ,$$

where the inequality relied on the fact that β (and hence also η) is sufficiently small. ◀

We are now ready to prove Proposition 28.

Proof of Proposition 28. Let $F : \{0,1\}^n \to \{0,1\}$ be a depth-2 formula of width w and size m. Let $F^{\mathtt{low}} : \{0,1\}^n \to \{0,1\}$ and $F^{\mathtt{up}} : \{0,1\}^n \to \{0,1\}$ be the β-lower-sandwiching and the β-upper-sandwiching formulas for F from Theorem 24, respectively, where $\beta = \frac{\alpha^6 \cdot (\delta/4)^4}{m^4 \cdot \log^6(1/\delta)}$. Note that the width of $F^{\mathtt{low}}$ and of $F^{\mathtt{up}}$ is at most w, and that their size is at most $2^{\tilde{O}(w) \cdot \log\log(m/\alpha\delta)}$.

According to Fact 14, the distribution of strings $\mathbf{r}$ over $\{0,1\}^{O(\log(w)) \cdot n}$, which is obtained by combining $\mathbf{y}$ and $\mathbf{z}$ and represents the pseudorandom restriction $\rho = \rho_{\mathbf{y},\mathbf{z}}$, is $(2 \cdot \delta')$-almost t'-wise independent. Hence, relying on Proposition 26, with probability at least $1 - 2\delta$ it holds both that $F^{\mathtt{low}}{\restriction_\rho}$ and $F^{\mathtt{up}}{\restriction_\rho}$ can be computed by decision trees of depth D, and that ρ keeps at least $\Omega(n/w)$ variables alive.

According to Theorem 25, all DNFs of width w are β-fooled by the distribution $\mathbf{z}$. [13] Therefore, relying on Lemma 29, for any fixed choice of $y \sim \mathbf{y}$, with probability at least $1 - 2\delta$

[13] Theorem 25 requires that the distribution $\mathbf{z}$ will be δ''-almost t''-wise independent, where $t'' = O(w^2 \cdot \log(w) + w \cdot \log(1/\beta)) = \tilde{O}(w) \cdot \log(m/\alpha\delta) < t'$ and $\log(1/\delta'') = O(w^2 \cdot \log^2(w) + w \cdot \log(w) \cdot \log(1/\beta)) = \tilde{O}(w) \cdot \log(m/\alpha\delta) < \log(1/\delta')$.

over $z \sim \mathbf{z}$ it holds that both $F^{\mathtt{low}}\restriction_\rho$ and $F^{\mathtt{up}}\restriction_\rho$ are α-close to $F\restriction_\rho$. Thus, the probability over choice of both $\mathbf{y}$ and $\mathbf{z}$ that $F^{\mathtt{low}}\restriction_\rho$ and $F^{\mathtt{up}}\restriction_\rho$ are α-close to $F\restriction_\rho$ is at least $1-2\delta$. ◀

5.2.3 Proofs of Theorems 2 and 3

We are now ready to prove Theorem 3. Recall that Theorem 3 asserts the existence of a hitting-set generator that is parametrized by a parameter $t > 0$.

▶ **Theorem 30.** *(Theorem 3, restated). Let $d \geq 2$, let $m : \mathbb{N} \to \mathbb{N}$ such that $m(n) \leq \mathrm{poly}(n)$, and let $t : \mathbb{N} \to \mathbb{N}$ such that $c_0 \leq t(n) \leq 2 \cdot \log(m(n))$, where c_0 is a sufficiently large constant. For every $n \in \mathbb{N}$, let $\mathcal{C}_n$ be the class of circuits $C : \{0,1\}^n \to \{0,1\}$ of size $m = m(n)$ and of depth at most d that accept all but at most $B(n)$ of their inputs, where $\log(B(n)) = \Omega\left(n^{1-1/\Omega(t)}/t^{d-2}\right)$ and $t = t(n)$. Then, there exists a hitting-set generator for $\mathcal{C} = \cup_{n\in\mathbb{N}}\mathcal{C}_n$ with seed length $\ell = \ell(n) = \tilde{O}\left(t^2 \cdot \log(n)\right)$.*

Theorem 2 follows as a corollary of Theorem 30, by using the specific parameter value $t = 2 \cdot \log(m)$, in which case $B(n) = 2^{\Omega\left(n/\log^{d-2}(n)\right)}$ and the seed length is $\tilde{O}\left(\log^3(n)\right)$.

Proof. Given input 1^n and a random seed in $\{0,1\}^\ell$, the hitting-set generator works in two steps. In the first step, the generator outputs a restriction $\bar{\rho} \in \{0,1,\star\}^n$ such that for any circuit C over n input bits of depth d and size $m = m(n)$, with high probability it holds that there exists a depth-2 formula C' of size $\mathrm{poly}(n)$ and width t that is both $(1/2)$-close to $C\restriction_{\bar{\rho}}$ and lower-sandwiching for $C\restriction_{\bar{\rho}}$. Moreover, with high probability the restriction $\bar{\rho}$ keeps at least $\log(B(n)) + 2$ variables alive.

Since the subcube $\mathfrak{C}(\bar{\rho})$ contains at least $4 \cdot B(n)$ inputs, the acceptance probability of $C\restriction_{\bar{\rho}}$ is at least $3/4$. Hence, the acceptance probability of C' is at least $1/4$ (because C' is $(1/2)$-close to $C\restriction_{\bar{\rho}}$), and every satisfying input for C' is also satisfying for C (because C' is lower-sandwiching for $C\restriction_{\bar{\rho}}$). Thus, in the second step, we use a pseudorandom generator for depth-2 circuits to "fool" C': The pseudorandom generator outputs a satisfying input for C' in $\mathfrak{C}(\bar{\rho})$ with positive probability, and any such input yields a satisfying input for C.

Parameter settings. Let $\epsilon > 0$ be a sufficiently small constant, and let $\delta = (\epsilon/m)$. Let $D = O(\log(1/\delta)) > 2 \cdot \log(2m/\delta)$, and let $m' = m \cdot 2^D = \mathrm{poly}(n)$. Let $\beta = \left(\frac{\delta}{2dm}\right)^{10^{2d}}$; we will use β as the approximation parameter whenever using Theorem 24. Let $\delta' > 0$ and $t' \in \mathbb{N}$ such that $\log(1/\delta') = O(t') = \tilde{O}\left(t^2 \cdot \log(n)\right)$.

The pseudorandom choice of restrictions. The algorithm that we will describe below constructs a sequence of restrictions. We mention in advance that when describing the algorithm, whenever we will say that we choose a restriction with a parameter $p = 2^{-q}$, the pseudorandom choice of restriction is the following:

- Let $\mathbf{y}$ be a distribution over $\{0,1\}^{\log(1/p)\cdot n}$ that is δ'-almost t'-wise independent.
- Let $\mathbf{z}$ be a distribution over $\{0,1\}^n$ that is δ'-almost t'-wise independent.
- The restriction $\rho = \rho_{\mathbf{y},\mathbf{z}}$ is chosen by sampling $y \sim \mathbf{y}$ in order to determine which variables are kept alive, and independently sampling $z \sim \mathbf{z}$ in order to determine values for the fixed variables.

Note that such a restriction keeps every variable alive with probability approximately p (i.e., with probability $p \pm \delta'$). The above process yields a distribution $\mathbf{r}$ over $\{0,1\}^{(\log(1/p)+1)\cdot n}$, which is obtained by combining $\mathbf{y}$ and $\mathbf{z}$ as detailed in the beginning of Section 5.2.1; according to Fact 14, the distribution $\mathbf{r}$ is $(2 \cdot \delta')$-almost t'-wise independent.

The first step. The generator constructs the restriction $\bar{\rho}$ as the composition of $2d-2$ retrictions $\bar{\rho} = \rho^{(2d-3)} \circ \rho^{(2d-4)} \circ ... \circ \rho^{(1)} \circ \rho^{(0)}$. The initial restriction $\rho^{(0)}$ is chosen with parameter $p = 1/O(1)$, and with probability $1-\epsilon$ it reduces the bottom fan-in of the circuit to $D = O(\log(1/\delta))$. [14] The next $2 \cdot (d-2)$ restrictions are applied in $d-2$ iterations. Loosely speaking, in each iteration, we apply a restriction that reduces the bottom fan-in to t, then define an approximating circuit (by replacing the formulas in the next-to-bottom layer, which have small width at this point, with small lower-sandwiching refinements, using Theorem 24), and finally apply a second restriction in order to "switch" the formulas in the next-to-bottom layer of the approximating circuit, and reduce the depth of the circuit.

Let $C^{(0)} = C{\restriction}_{\rho^{(0)}}$ be the circuit in the beginning of the first iteration, and note that $C^{(0)}$ is of depth d, size at most $m < m'$, and bottom fan-in at most D. For $i \in [d-2]$, let us describe the i^{th} iteration. Assuming all previous iterations were successful, in the beginning of the i^{th} iteration we start with a circuit $C^{(i-1)}$ of depth at most $d-(i-1)$, bottom fan-in at most D, and with at most $m' = m \cdot 2^D$ gates in its bottom layer. We will produce two restrictions, denoted $\rho^{(2i-1)}$ and $\rho^{(2i)}$, and define a circuit $C^{(i)}$ whose domain is $\mathfrak{C}(\rho^{(2i)} \circ \rho^{(2i-1)} \circ ... \circ \rho^{(0)})$ such that with probability $1-O(\epsilon)$ it holds that $C^{(i)}$ is of depth at most $d-i$, bottom fan-in D, and the number of gates in its bottom layer is at most m'. (After we finish the description of a single iteration, we will also prove that for any $i \in [d-2]$ it holds that $C^{(i)}{\restriction}_{\bar{\rho}}$ is close to $C^{(i-1)}{\restriction}_{\bar{\rho}}$; see Claim 30.2 below.)

The first restriction in iteration i, denoted $\rho^{(2i-1)}$, is chosen with the parameter $p = \left(\epsilon / \left(m \cdot 2^{2D+1}\right)\right)^{1/t} = n^{-1/\Omega(t)}$. We now show that with probability at least $1-O(\epsilon)$ the bottom fan-in of the circuit $C^{(i-1)}{\restriction}_{\rho^{(2i-1)}}$ is less than t. To do so, first note the following:

▶ **Claim 30.1.** *Let S be a fixed set of at most D variables. Then, with probability at least $1-\epsilon/m'$ it holds that less than t variables in S are kept alive by $\rho^{(2i-1)}$.*

Proof. Recall that the restriction $\rho^{(2i-1)}$ is chosen such that the distribution $\mathbf{y}$ over $\{0,1\}^{\log(1/p)\cdot n}$, which determines which variables will be kept alive, is δ'-almost t'-wise independent. We will only need the fact that the blocks of size $\lceil \log(1/p) \rceil$ in $\mathbf{y}$ are (p^t)-almost t-wise independent; this holds because $t \cdot \lceil \log(1/p) \rceil < O(\log(m/\epsilon)) < t'$, and $\delta' < p^t = 1/\mathrm{poly}(n)$.

For any fixed set of t variables in S, the probability that all variables in the set remain alive after applying a uniformly-chosen restriction with the parameter p is p^t. Since the blocks of size $\lceil \log(1/p) \rceil$ in $\mathbf{y}$ are (p^t)-almost t-wise independent, the probability that $\rho^{(2i-1)}$ keeps all t variables alive is at most $2 \cdot (p^t)$. Thus, the probability that $\rho^{(2i-1)}$ keeps t variables in S alive is at most $\binom{|S|}{t} \cdot 2 \cdot p^t < 2^{D+1} \cdot p^t < \epsilon/m'$. ◀

Recall that the number of gates in the bottom layer of $C^{(i-1)}$ is at most m', and that each of them is of fan-in at most D. Using Claim 30.1 and a union-bound, with probability at least $1-\epsilon$ it holds that the bottom fan-in of $C^{(i-1)}{\restriction}_{\rho^{(2i-1)}}$ is less than t.

Assuming that the bottom fan-in of $C^{(i-1)}{\restriction}_{\rho^{(2i-1)}}$ is indeed less than t, we now use Theorem 24 to replace each formula F in the next-to-bottom layer of $C^{(i-1)}{\restriction}_{\rho^{(2i-1)}}$ with a

[14] To see that such a restriction indeed reduces the bottom fan-in, fix a gate in the bottom layer of fan-in more than $2 \cdot \log(2m/\epsilon)$. The probability under a uniformly-chosen restriction with $p = 1/4$ that none of the lexicographically-first $2 \cdot \log(2m/\epsilon)$ variables feeding into the gate is fixed to a satisfying value is $\left(\frac{1+p}{2}\right)^{2 \cdot \log(2m/\epsilon)} < \epsilon/2m$. Since this event depends only on the values that the restriction assigns to $2 \cdot \log(2m/\epsilon)$ variables, and the value for each variable depends on $\log(1/p) = O(1)$ bits, the event depends on at most $O(\log(m/\epsilon))$ bits of the restriction. Thus, the event happens with probability at most ϵ/m when the restriction is chosen from a $1/\mathrm{poly}(m/\epsilon)$-biased set.

Table 1 Summary of the restrictions that are applied in the first step.

	Value of p	**Goal of the restriction**
$\rho^{(0)}$	$1/O(1)$	Reduce the bottom fan-in to D
$i = 1, ..., d-2$:		
$\rho^{(2i-1)}$	$n^{-1/\Omega(t)}$	Reduce the bottom fan-in to t
$\rho^{(2i)}$	$1/O(t)$	"Switch" the width-t formulas at the next-to-bottom-layer
$\rho^{(2d-3)}$	$n^{-1/\Omega(t)}$	Reduce the bottom fan-in to t

β-lower-sandwiching refinement $F^{\mathtt{low}}$ such that the size of $F^{\mathtt{low}}$ is at most $2^{\tilde{O}(t)\cdot\log\log(1/\beta)}$. Let $\widetilde{C^{(i-1)}\restriction_{\rho^{(2i-1)}}}$ be the circuit that is obtained by replacing all the formulas in the next-to-bottom layer of $C^{(i-1)}\restriction_{\rho^{(2i-1)}}$ in this manner.

The final step in the i^{th} iteration is to apply a restriction $\rho^{(2i)}$ with parameter $p = 1/O(t)$ that is intended to simplify each formula $F^{\mathtt{low}}$ in the next-to-bottom layer of $\widetilde{C^{(i-1)}\restriction_{\rho^{(2i-1)}}}$ to a decision tree of depth at most D. Let $C^{(i)} = \left(\widetilde{C^{(i-1)}\restriction_{\rho^{(2i-1)}}}\right)\restriction_{\rho^{(2i)}}$. Relying on Proposition 26, the restriction $\rho^{(2i)}$ is successful with probability at least $1 - O(\epsilon)$, and in this case the circuit $C^{(i)}$ is of depth at most $d - i$, and the bottom layer of $C^{(i)}$ has at most $m' = m \cdot 2^D$ gates, each of fan-in at most D. [15]

We now apply one final restriction $\rho^{(2d-3)}$, with parameter $p = \left(\epsilon/\left(m \cdot 2^{2D+1}\right)\right)^{1/t}$, in order to reduce the bottom fan-in of $C^{(d-2)}$ to t. Using Claim 30.1 and a union-bound, with probability at least $1 - O(\epsilon)$ it holds that the width of $C^{(d-2)}\restriction_{\rho^{(2d-3)}}$ is at most t. For convenience, in Table 1 we summarize the restrictions that were applied in the first step.

Let $C^{(d-1)} = C^{(d-2)}\restriction_{\rho^{(2d-3)}}$, and recall that $\bar{\rho} = \rho^{(2d-3)} \circ \rho^{(2d-2)} \circ ... \circ \rho^{(0)}$. The above shows that if all the iterations are successful, then $C^{(d-1)}$ is a formula of depth 2, size at most m', and width t. Also note that if all the iterations are successful, then $\widetilde{C^{(d-1)}}$ is lower-sandwiching for $C\restriction_{\bar{\rho}}$. This is because for every $i \in [d-2]$ it holds that $\widetilde{C^{(i-1)}\restriction_{\rho^{(2i-1)}}}$ is lower-sandwiching for $C^{(i-1)}\restriction_{\rho^{(2i-1)}}$ (since $\widetilde{C^{(i-1)}\restriction_{\rho^{(2i-1)}}}$ is obtained by replacing every formula F in the next-to-bottom-layer of $C^{(i-1)}\restriction_{\rho^{(2i-1)}}$ with a lower-sandwiching refinement $F^{\mathtt{low}}$), which implies that $C^{(i)}\restriction_{\bar{\rho}} = \left(\widetilde{C^{(i-1)}\restriction_{\rho^{(2i-1)}}}\right)\restriction_{\bar{\rho}}$ is lower-sandwiching for $C^{(i-1)}\restriction_{\bar{\rho}}$.

The main thing that is left to prove in the analysis of the first step is that with probability at least $1 - O(\epsilon)$ it holds that $C^{(d-1)}$ is $(1/2)$-close $C\restriction_{\bar{\rho}}$. To do so, we will show that with probability at least $1 - O(\epsilon)$, for every $i \in [d-2]$ it holds that $C^{(i-1)}\restriction_{\bar{\rho}}$ is $(1/2d)$-close to $C^{(i)}\restriction_{\bar{\rho}}$. Assuming that the latter holds, we can deduce that $C\restriction_{\bar{\rho}} = C^{(0)}\restriction_{\bar{\rho}}$ is $1/2$-close to $C^{(d-1)} = C^{(d-2)}\restriction_{\bar{\rho}}$. Thus, it suffices to prove the following claim:

▶ **Claim 30.2.** *For any $i \in [d-2]$, with probability at least $1 - O(\epsilon)$ it holds that $C^{(i)}\restriction_{\bar{\rho}}$ is $(1/2d)$-close to $C^{(i-1)}\restriction_{\bar{\rho}}$.*

[15] Specifically, we rely on Proposition 26 with width parameter t, error parameter δ, size parameter $2^{\tilde{O}(t)\cdot\log\log(1/\beta)}$, and depth bound D for the decision trees. Proposition 26 requires that the distribution $\mathbf{r}$ of restrictions will be δ''-almost t''-wise independent, where $\log(1/\delta'') = O(t'') = \tilde{O}(t^2)\cdot\log(1/\delta)\cdot\log\log(1/\beta) = \tilde{O}(t^2\cdot\log(n))$. The latter holds by our choice of δ' and t'.

Proof. Let $i \in [d-2]$, let F be a formula in the next-to-bottom layer of $C^{(i-1)}\restriction_{\rho^{(2i-1)}}$, and let $F^{\mathtt{low}}$ be a β-refinement of F. We will prove that with probability $1-O(\delta)$ it holds that $F^{\mathtt{low}}\restriction_{\bar{\rho}}$ is $(1/2dm)$-close to $F\restriction_{\bar{\rho}}$. This suffices to prove Claim 30.2, since by a union-bound over m formulas it follows that with probability at least $1-O(\epsilon)$ it holds that the circuit $\left(\widetilde{C^{(i-1)}\restriction_{\rho^{(2i-1)}}}\right)\restriction_{\bar{\rho}} = C^{(i)}\restriction_{\bar{\rho}}$ is $(1/2d)$-close to $\left(C^{(i-1)}\restriction_{\rho^{(2i-1)}}\right)\restriction_{\bar{\rho}} = C^{(i-1)}\restriction_{\bar{\rho}}$.

For every $j \in \{2i, ..., 2d-3\}$, let $\rho^{(2i,...,j)}$ be the composed restriction $\rho^{(2i,...,j)} = \rho^{(j)} \circ ... \circ \rho^{(2i)}$, and let $\beta_j = (\delta/2dm)^{10^{2d-3-j}}$. We will prove the following statement: For every $j \in \{2i, ..., 2d-3\}$, with probability at least $1-O(\delta)$ it holds that $F^{\mathtt{low}}\restriction_{\rho^{(2i,...,j)}}$ is a β_j-refinement of a depth-2 formula of size m' and width t for $F\restriction_{\rho^{(2i,...,j)}}$. Invoking this statement with $j = 2d-3$, we can deduce that with probability at least $1-O(\delta)$ it holds that $F^{\mathtt{low}}\restriction_{\bar{\rho}}$ is β_{2d-3}-close to $F\restriction_{\bar{\rho}}$, where $\beta_{2d-3} < 1/2dm$.

We prove the aforementioned statement by induction on j. For the base case $j = 2i$, we start with a formula F of size m' and width t, and a β-refinement $F^{\mathtt{low}}$ of F, where $\beta < \beta_0 \leq \beta_{j-1}$. Now, $\rho^{(j)}$ is chosen according to a distribution such that for every fixed choice of variables to keep alive (i.e., every fixed $y \sim \mathbf{y}$), the choice of values for the fixed variables (i.e., $z \sim \mathbf{z}$) is δ'-almost t'-wise independent. Relying on Theorem 25 and on our choice of δ' and t', the distribution distribution $\mathbf{z}$ β-fools all DNFs of width w. We can therefore rely on Lemma 29 to deduce that with probability at least $1-O(\delta)$ it holds that $F^{\mathtt{low}}\restriction_{\rho^{(j)}}$ is a β_j-refinement of $F\restriction_{\rho^{(j)}}$. [16]

The induction step, for $j \geq 2i+1$, is very similar to the base case. By the induction hypothesis, with probability at least $1-O(\delta)$ it holds that $F^{\mathtt{low}}\restriction_{\rho^{(2i,...,j-1)}}$ is a (β_{j-1})-refinement of a size m' and width w' depth-2 formula for $F\restriction_{\rho^{(2i,...,j-1)}}$. We can then use Theorem 25 and Lemma 29 similarly to the base case. ◀

To conclude the analysis of the first step, note that with probability at least $1-O(\epsilon)$ it holds that at least $\log(B(n))+2 = \Omega\left(n^{1-1/\Omega(t)}/t^{d-2}\right)$ variables remain alive. To see that this is the case, recall that $\bar{\rho}$ is comprised of one restriction with parameter $p_0 = 1/O(1)$, and $d-1$ restrictions with parameter $p_1 = n^{-1/\Omega(t)}$, and $d-2$ restrictions with parameter $p_2 = 1/O(t)$. Let $\bar{p} = p_0 \cdot p_1^{d-1} \cdot p_2^{d-2} \cdot n$, and note that $\bar{p} = \Omega\left(n^{1-1/\Omega(t)}/t^{d-2}\right)$.

The expected number of living variables under $\bar{\rho}$ is $\Theta(\bar{p})$ (because in each restriction with parameter p, every variable is kept alive with probability $p \pm O(\delta') \in p \pm (p/2)$). Since all the choices of variables to keep alive are according to distributions that are δ'-almost t'-wise independent, we can use Fact 13 to deduce that with probability at least $1-O(\epsilon)$ it holds that at least $\Omega(\bar{p}) = \Omega\left(n^{1-1/\Omega(t)}/t^{d-2}\right) > \log(B(n))+2$ variables remain alive after the first step. (When using Fact 13, we relied on the fact that t is larger than a sufficiently large constant c_0 to deduce that $n^{1-1/\Omega(t)}/t^{d-2} > n^{\Omega(1)}$).

The second step. We now invoke the pseudorandom generator from Theorem 25 for depth-2 circuits of width t, instantiated with error parameter $1/8$, and output the string that the generator outputs, completed to a string of length n according to $\bar{\rho}$. The generator requires a seed of length $O(t^2 \cdot \log^2(t)) = \tilde{O}(t^2)$.

[16] We invoke Lemma 29 with width parameter t, size bound m', and error parameter δ. We know that $F^{\mathtt{low}}$ is a β_{j-1}-refinement of F, and we want to deduce that with probability at least $1-O(\delta)$ it holds that $F^{\mathtt{low}}\restriction_{\rho^{(j)}}$ is an α-refinement of $F\restriction_{\rho^{(j)}}$, where $\alpha = \beta^j$. The lemma requires that the distribution $\mathbf{z}$ will (β_{j-1})-fool all DNFs of width t, and that $\beta_{j-1} \leq \frac{\beta_j^6 \cdot (\delta/4)^4}{m^4 \cdot \log^6(1/\delta)}$, both of which indeed hold.

Let us now prove this yields a satisfying input for C, with positive probability. If the first step was successful, then $\bar{\rho}$ kept more than $\log(B(n)) + 2$ live variables, and hence the acceptance probability of $C\restriction_{\bar{\rho}}$ is at least $3/4$. Since $C^{(d-1)}$ is $1/2$-close to $C\restriction_{\bar{\rho}}$, it follows that $\Pr_{x \in \mathfrak{C}(\bar{\rho})}[C^{(d-1)}(x) = 1] \geq 1/4$. Thus, the generator outputs a satisfying input for $C^{(d-1)}$, with positive probability, and this input (when completed to a string of length n according to $\bar{\rho}$) is satisfying for C, because $C^{(d-1)}$ is lower-sandwiching for $C\restriction_{\bar{\rho}}$. ◀

6 Constant-depth circuits with parity gates

In this section we prove the claims made in Section 1.3: In Section 6.1 we prove Theorem 5, and in Section 6.2 we prove Theorem 6.

6.1 Proof of Theorem 5

The proof is similar to the proof of Theorem 1, and is a variation on [9, Thm 4.2 and Remark 4.4]. Starting from a CNF C, we will employ error-reduction within $\mathcal{AC}^0[\oplus]$, by first sampling inputs for C using Trevisan's extractor [22], and then taking the disjunction of the evaluation of C on these inputs (rather than an approximate majority, as in [9]). This will yield a layered circuit of the form $\vee \wedge \vee \oplus$ that accepts all but 2^{n^c} of its inputs, for any desired $c > 0$. Details follow.

Let $C : \{0,1\}^n \to \{0,1\}$ be a CNF that accepts most of its inputs. For $n' = n^{(1/c)+1}$ and $s = O(\log(n))$, let $E : \{0,1\}^{n'} \times \{0,1\}^s \to \{0,1\}^n$ be Trevisan's extractor instantiated for min-entropy $(n')^c = n^{1+\Omega(1)}$ and error parameter $1/4$. We construct a circuit $C' : \{0,1\}^{n'} \to \{0,1\}$ that first computes the values $E(x,z)$, for each possible seed $z \in \{0,1\}^s$, then evaluates C on each value $E(x,z)$, and finally takes an OR of these evaluations; that is, $C'(x) = \vee_{z \in \{0,1\}^s} C\left(E(x,z)\right)$.

Note that C' is a layered depth-4 circuit of the form $\vee \wedge \vee \oplus$, since for each seed $z \in \{0,1\}^s$, the residual function $E_z(x) = E(x,z)$ is just a linear transformation of x. Also note that the number of inputs $x \in \{0,1\}^{n'}$ for which $\Pr_z[C(E(x,z))] < 1/4$ is at most $2^{(n')^c}$. In particular, C' accepts all but at most $2^{(n')^c}$ of its inputs, and for each satisfying input x for C', we can find a corresponding satisfying input for C among $\{E(x,z)\}_{z \in \{0,1\}^s}$.

6.2 Proof of Theorem 6

The current section is organized as follows. In Section 6.2.1 we present two algorithmic tools that will be used in the proof: An adaptation of the approach of Chaudhuri and Radhakrishnan [4] to the setting of $\oplus \wedge \oplus$ circuits, and an adaptation of Viola's pseudorandom generator [25] to polynomials that are defined over an affine subspace. Then, in the next three sections, we prove the corresponding three items of Theorem 6.

We rely on the notion of *affine restrictions*. A restriction of a circuit $C : \{0,1\}^n \to \{0,1\}$ to an affine subspace $W \subseteq \{0,1\}^n$ will be constructed by accumulating a list of (independent) affine conditions that defines W. That is, each of the various algorithms will construct a full-rank matrix A and a vector b such that $W = \{x : Ax = b\}$. For an affine function g, when we say that an algorithm "adds $g = 0$ to the list of affine conditions", we mean that it extends A by adding the linear part of g as an additional row to A, and extends b by adding the constant term of g as an additional bit to b (i.e., if $g(x) = \sum_{i=1}^n c_i x_i + c_0$ then the row $c = (c_1, ..., c_n)$ is added to A and c_0 is added to b). After each addition of a condition, we will say that the algorithm "simplifies the circuit accordingly"; by this we mean that for any $\oplus$-gate g' in the bottom layer whose linear function is dependent on the rows of A, the

algorithm fixes g' to the appropriate value determined by A and b, and, if g' was fixed to zero, then the algorithm removes all the $\wedge$-gates that g' feeds into.

6.2.1 Two algorithmic tools

Let us first adapt the approach of Chaudhuri and Radhakrishnan [4], which was originally used to construct "bit-fixing" restrictions for $\mathcal{AC}^0$ circuits, to the setting of $\oplus \wedge \oplus$ circuits and affine restrictions.

▶ **Proposition 31** (Whitebox Affine Restrictions for $\oplus \wedge \oplus$ Circuits). *For two integers $m_\wedge$ and $m_\oplus$, let $\mathcal{C}$ be the class of $\oplus \wedge \oplus$ circuits over n input bits with $m_\wedge$ gates in the middle layer and $m_\oplus$ gates in the bottom layer. Then, for any two integers $d_\oplus$ and $d_\wedge$, there exists a polynomial-time algorithm that, when given as input a circuit $C \in \mathcal{C}$, outputs an affine subspace $W \subseteq \{0,1\}^n$ such that:*
1. *In the restriction of C to W, each $\wedge$-gate in the middle layer has fan-in at most $d_\wedge$.*
2. *The subspace W is of co-dimension at most $\frac{m_\wedge}{d_\oplus} + \frac{d_\oplus \cdot m_\oplus}{d_\wedge}$.*

Proof. The algorithm operates in two steps. In the first step, as long as there exists a $\oplus$-gate g in the bottom layer with *fan-out* at least $d_\oplus$, the algorithm adds the condition $g = 0$ to the list of affine conditions, and simplifies the circuit accordingly. Note that each addition of a condition as above fixes at least $d_\oplus$ of the $\wedge$-gates in the middle layer, and thus at most $m_\wedge/d_\oplus$ conditions are added (or else the entire circuit simplifies to a constant). Hence, after the first step concludes, the fan-out of each $\oplus$-gate in the bottom layer is $d_\oplus$, and at most $m_\wedge/d_\oplus$ affine conditions have been accumulated.

In the second step, as long as there exists an $\wedge$-gate g in the middle layer with *fan-in* at least $d_\wedge$, the algorithm (arbitrarily) chooses one $\oplus$-gate g' that feeds into g, adds the condition $g' = 0$ to the list of affine conditions, and simplifies the circuit accordingly. Note that, in the beginning of the second step, the number of wires feeding the middle layer is at most $d_\oplus \cdot m_\oplus$ (since there are at most $m_\oplus$ gates in the bottom layer, each of them with fan-out at most $d_\oplus$). Now, note that each addition of an affine condition in the second step eliminates at least $d_\wedge$ wires; thus, the algorithm adds at most $\frac{d_\oplus}{d_\wedge} \cdot m_\oplus$ conditions in the second step. After the second step is complete, each $\wedge$-gate in the middle layer has fan-in at most $d_\wedge$, and the list of affine conditions contains at most $m_\wedge/d_\oplus + \frac{d_\oplus}{d_\wedge} \cdot m_\oplus$ conditions. ◀

We now verify that we can use Viola's pseudorandom generator [25] in order to "fool" $\oplus \wedge \oplus$ circuits that, when restricted to an affine subspace, have a constant maximal fan-in of the $\wedge$-gates.

▶ **Proposition 32** (Invoking Viola's PRG in an Affine Subspace). *There exists an algorithm G that, for every $n \in \mathbb{N}$, when G is given as input an integer D, a seed of $\ell = O(\log(n))$ bits, and a basis for an affine subspace $W \subseteq \{0,1\}^n$, then G runs in time $\mathrm{poly}(n)$ and satisfies the following: For every $\oplus \wedge \oplus$ circuit C over n input bits such that C simplifies under the restriction W to a $\oplus \wedge \oplus$ circuit in which the maximal fan-in of $\wedge$-gates is D and such that $C\restriction_W \not\equiv 0$, it holds that $\Pr[C(G(\boldsymbol{u}_\ell)) = 1] > 0$.*

Proof. Denote the dimension of W by $m = \dim(W)$. The algorithm G first finds a full-rank $n \times m$ matrix B and $s \in \{0,1\}^n$ such that $x \mapsto Bx + s$ maps $\{0,1\}^m$ to W. Then, the algorithm G uses its random seed to invoke Viola's pseudorandom generator for polynomials $\mathbb{F}_2^m \to \mathbb{F}_2$ of degree D, with error parameter $2^{-(D+1)}$, thus obtaining a string $x \in \{0,1\}^m$. Finally, the algorithm G outputs the string $Bx + s$.

Now, let C be $\oplus\wedge\oplus$ circuit as in the hypothesis, and consider the polynomial $p : \mathbb{F}_2^m \to \mathbb{F}_2$ such that $p(x) = C(Bx + s)$. Note that p is of degree D, because C computes an sum of monomials of degree D over $\mathbb{F}_2$, and the affine transformation does not increase the degree. Also, using our hypothesis that p is non-zero, it follows that the acceptance probability of p is at least 2^{-D}. Thus, the probability that Viola's generator will output x such that $p(x) = 1$ is at least $2^{-(D+1)} > 0$, and each such x yields a string $y = Bx + s$ such that $C(y) = 1$. ◀

6.2.2 Linear-sized circuits with $B(n) = 2^{-\Omega(n)}$

We prove the first item of Theorem 6 by invoking the whitebox algorithm from Proposition 31 with appropriate parameters $d_\wedge, d_\oplus = O(1)$, and then using the generator from Proposition 32.

▶ **Proposition 33** (Theorem 6, Item (1): Hitting Biased Linear-Sized $\oplus\wedge\oplus$ Circuits)**.** *Let $\epsilon > 0$ be an arbitrarily small constant, and let $c > 0$ be an arbitrarily large constant. Let $\mathcal{C}$ be the class of $\oplus\wedge\oplus$ circuits such that any circuit $C \in \mathcal{C}$ over n input bits has at most $c \cdot n$ gates and accepts all but at most $2^{(1-\epsilon)\cdot n}$ of its inputs. Then, there exists a polynomial-time algorithm that, when given any circuit $C \in \mathcal{C}$, finds a satisfying input for C.*

Proof. The algorithm first invokes the algorithm from Proposition 31 with parameters $d_\oplus = \frac{4\cdot c}{\epsilon}$ and $d_\wedge = d_\oplus^2$, to obtain an affine subspace W of co-dimension at most

$$\frac{m_\wedge}{d_\oplus} + \frac{d_\oplus \cdot m_\oplus}{d_\wedge} < 2 \cdot \frac{c \cdot n}{(4 \cdot c)/\epsilon} = \frac{\epsilon}{2} \cdot n$$

such that in the restriction of C to W, every $\wedge$-gate in the middle layer has fan-in at most $d_\wedge = O(1)$. Since the circuit C has at most $2^{(1-\epsilon)\cdot n}$ unsatisfying inputs, it follows that $\Pr_{w\in W}[C(w) = 1] \geq 1 - 2^{-(\epsilon/2)\cdot n}$. Thus, the algorithm concludes by invoking the algorithm from Proposition 32. ◀

6.2.3 Sub-quadratic circuits with $(1 + o(1)) \cdot n$ bottom $\oplus$-gates and $B(n) = 2^{n^c}$

We now prove the second item of Theorem 6.

▶ **Proposition 34** (Theorem 6, Item (2): Hitting Biased Sub-quadratic $\oplus\wedge\oplus$ Circuits)**.** *Let $\epsilon > 0$ and let $0 < c < \epsilon$. Let $\mathcal{C}$ be the class of $\oplus\wedge\oplus$ circuits such that any $C \in \mathcal{C}$ over n input bits has at most $n + n^c$ bottom $\oplus$-gates, and at most $n^{2-\epsilon}$ middle $\wedge$-gates, and accepts all but $B(n) = 2^{n^c}$ of its inputs. Then, there exists a polynomial-time algorithm that, when given any circuit $C \in \mathcal{C}$, finds a satisfying input for C.*

Proof. Recall that a high-level overview of the proof, which used the parameter values $m_\wedge = n^{1.1}$ and $m_\oplus = n$, appeared in Section 2.3. Let us first explain, in high-level, how to handle the setting of $m_\wedge \leq n^{2-\epsilon}$; for the moment, we are still assuming that $m_\oplus = n$. As in the overview in Section 2.3, the algorithm works in two steps. In the first step, we use Proposition 31 to fix $o(m_\oplus)$ of the $\oplus$-gates such that after the restriction, the fan-in of the $\wedge$-gates is bounded by $w = n^{1-\alpha\cdot\epsilon}$, where $\alpha < 1$ is a constant slightly smaller than 1; this is possible because $m_\wedge \leq n^{2-\epsilon}$ (see the proof details below). In the second step, we restrict the $\oplus$-gates using an $O(1)$-independent distribution, keeping each $\oplus$-gate alive with probability $p = n^{-(1-\beta\cdot\epsilon)}$, where $\beta < \alpha$ (and recall that we choose arbitrary consistent values for the gates that are fixed). The crucial point is the following: On the one hand, since $p \leq 1/w^{1+\Omega(1)}$, after the second step the fan-in of the $\wedge$-gates is upper-bounded by a constant (as explained in Section 2.3); and on the other hand, the number of living $\oplus$-gates

after the second step is approximately $p \cdot (1 - o(1)) \cdot n = \Omega\left(n^{\beta \cdot \epsilon}\right) > n^c = \log(B(n))$, where the inequality holds if we choose $\beta > c/\epsilon$ (which is possible if we initially choose $\alpha \in (c/\epsilon, 1)$).

To see how we handle the setting of $m_\oplus \leq n + n^c$ (rather than $m_\oplus = n$), note that the overall number of affine conditions that the algorithm imposes is $m_\oplus - \Omega(p \cdot m_\oplus)$. Since $m_\oplus \leq n + o(p \cdot n)$, the number of affine conditions is at most $n - \Omega(p \cdot n)$, which means that the affine subspace W is of dimension $\Omega(p \cdot n) > \log(B(n))$.

Let us now provide the full details for the proof. Assume, without loss of generality, that $m_\oplus \geq n$ (we can add dummy gates if necessary). We first invoke the algorithm from Proposition 31 with parameters $d_\wedge = n^{1-\alpha\cdot\epsilon}$, where $\alpha = \frac{(c/\epsilon)+1}{2}$, and $d_\oplus = n^{1-\alpha'\cdot\epsilon}$, where $\alpha' = (c/\epsilon) + (2/3) \cdot (1 - c/\epsilon) > \alpha$. The algorithm outputs an affine subspace of co-dimension at most

$$\begin{aligned}\frac{m_\wedge}{d_\oplus} + \frac{d_\oplus \cdot m_\oplus}{d_\wedge} &\leq n^{2-\epsilon-(1-\alpha'\cdot\epsilon)} + n^{1-\alpha'\cdot\epsilon-(1-\alpha\cdot\epsilon)} \cdot m_\oplus \\ &= n^{1-(1-\alpha')\cdot\epsilon} + n^{-(\alpha'-\alpha)\cdot\epsilon} \cdot m_\oplus \ ,\end{aligned}$$

which is $o(m_\oplus)$, such that in the restriction of C to the subspace, every $\wedge$-gate in the middle layer has fan-in at most $d_\wedge = n^{1-\alpha\cdot\epsilon}$.

Denote the number of $\oplus$-gates that were not fixed in the previous step by m', and consider the following pseudorandom restriction process. For a sufficiently large constant $\gamma > 1$ (which will be determined later), we use a γ-wise independent distribution over $[1/p]^{n'}$, where $p = n^{-(1-\beta\cdot\epsilon)}$ and $\beta = (c/\epsilon) + (1/3) \cdot (1 - c/\epsilon) < \alpha$. [17] Denote the random variable that is the output string of this distribution by $\rho \in [1/p]^{n'}$. For every $\oplus$-gate that has not been restricted by the algorithm from Proposition 31, the algorithm now marks the gate as "alive" if and only if the corresponding element in the string ρ equals zero; otherwise, it marks the gate as "fixed".

For any $\wedge$-gate g in the middle-layer, the probability that at least γ gates that feed into g are marked "alive" is at most

$$\binom{d_\wedge}{\gamma} \cdot p^\gamma < n^{(1-\alpha\cdot\epsilon)\cdot\gamma} \cdot n^{-(1-\beta\cdot\epsilon)\cdot\gamma} = n^{-(\alpha-\beta)\cdot\epsilon\cdot\gamma} \ ,$$

which can be made less than $1/m_\wedge = n^{-(2-\epsilon)}$ by an appropriate choice of γ (i.e., $\gamma > \frac{2-\epsilon}{(\alpha-\beta)\cdot\epsilon}$). After union-bounding over all $\wedge$-gates, we have that with probability at least 0.99, each $\wedge$-gate is fed by less than γ of the "alive" $\oplus$-gates. Also note that with probability at least 0.99, the number of $\oplus$-gates that were marked as "alive" is at least $(p \cdot m')/2$; this is because the distribution is γ-wise independent (so we can use Fact 12). The algorithm and finds a choice of ρ, denoted by ρ_0, that meets both these conditions (by enumerating the outputs of the γ-wise independent distribution). Then, the algorithm iteratively fixes values for the $\oplus$-gates that are marked as "fixed" by ρ_0. Specifically, as long as there is a $\oplus$-gate g that is marked as "fixed" by ρ_0, the algorithm adds the condition $g = 0$ to the list of affine conditions that defines W, and simplifies the circuit accordingly.

Let us now count the number of affine conditions that the algorithm imposed (i.e., the co-dimension of W). After all the restrictions, the number of living variables is at least $(p/2) \cdot m' \geq (p/2) \cdot (1 - o(1)) \cdot m_\oplus \geq (p/3) \cdot m_\oplus$, which implies that the number of affine

[17] We will actually use the value $p = 2^{-\lceil(1-\beta\cdot\epsilon)\cdot\log(n)\rceil}$, such that $1/p$ is a power of 2, but the difference between this value and $n^{-(1-\beta\cdot\epsilon)}$ is insignificant in what follows.

conditions is at most $m_\oplus - (p/3) \cdot m_\oplus$. Since $m_\oplus \leq n + n^c$, we have that

$$\begin{aligned} m_\oplus - (p/3) \cdot m_\oplus &< n + n^c - (p/3) \cdot n \\ &= n + n^c - \frac{1}{3} \cdot n^{\beta \cdot \epsilon} , \end{aligned}$$

which is less than $n - n^c$, because $n^c = o(n^{\beta \cdot \epsilon})$ (since $\beta \cdot \epsilon = c + \Omega(1)$).

Thus, the algorithm is left with a subspace W of dimension more than $n^c = \log(B(n))$ such that when the circuit C is restricted to the subspace W, the fan-in of every $\wedge$-gate in the middle layer is at most $\gamma = O(1)$. Hence, at this point the algorithm can invoke the algorithm from Proposition 32, and find a satisfying input for C in W. ◀

6.2.4 Circuits with a slightly super-linear number of bottom $\oplus$-gates and slightly sub-linear number of $\wedge$-gates

We now prove the third item of Theorem 6. The crucial observation here is that after invoking the algorithm from Proposition 31, the number of $\oplus$-gates is at most $m_\wedge \cdot d_\wedge$, since this is the number of wires that feed into the middle layer.

▶ **Proposition 35** (Theorem 6, Item (3): Hitting Biased $\oplus \wedge \oplus$ Circuits with a Super-Linear Number of $\oplus$-Gates)**.** *For any constant $\epsilon > 0$, let $\mathcal{C}$ be the class of $\oplus \wedge \oplus$ circuits such that any circuit $C \in \mathcal{C}$ over n input bits has at most $n^{1+\epsilon}$ gates in the bottom layer and at most $(1/5) \cdot n^{1-\epsilon}$ gates in the middle layer, and accepts all but at most $B(n) = 2^{n/15}$ of its inputs. Then, there exists a polynomial-time algorithm that, when given any circuit $C \in \mathcal{C}$, finds a satisfying input for C.*

Proof. We first invoke the algorithm from Proposition 31 with parameters $d_\oplus = 1$ and $d_\wedge = (5/2) \cdot n^\epsilon$. The algorithm outputs an affine subspace W' of co-dimension at most

$$\frac{m_\wedge}{d_\oplus} + \frac{d_\oplus \cdot m_\oplus}{d_\wedge} \leq (1/5) \cdot n^{1-\epsilon} + (2/5) \cdot n$$

such that in the restriction of C to W', every $\wedge$-gate in the middle layer has fan-in at most $d_\wedge = (5/2) \cdot n^\epsilon$. Since there are at most $m_\wedge = (1/5) \cdot n^{1-\epsilon}$ gates in the middle layer, it follows that there are at most $m_\wedge \cdot d_\wedge = n/2$ bottom $\oplus$-gates that influence the output of $C{\restriction}_{W'}$. By fixing values for these gates, we obtain a subspace W of dimension at least $(1/2 - (2/5) - o(1)) \cdot n > n/15$ such that $C{\restriction}_W$ is constant. Since $B(n) = 2^{n/15}$, it follows that $C{\restriction}_W \equiv 1$, and thus we can output any $w \in W$. ◀

7 Polynomials that vanish rarely

In the current section we prove Theorem 7 (in Section 7.1) and Theorem 8 (in Section 7.2). Recall that throughout the currnet section we consider a normalized "badness" parameter $b(n) = B(n)/2^n$.

7.1 Proof of Theorem 7

We now prove a more general version of Theorem 7, which depends on additional parameters; after stating this general version, we will spell out the parameter choices that yield Theorem 7. The proof relies on Lemma 16.

▶ **Proposition 36** (Theorem 7, Parametrized Version). *For $m : \mathbb{N} \to \mathbb{N}$ and $b : \mathbb{N} \to [0, \frac{1}{2}]$, let $\mathcal{C}$ be the class of $\oplus \wedge \oplus$ circuits over n input bits with $m = m(n)$ $\wedge$-gates that accept all but a $b(n)$ fraction of their inputs. For any $d \geq 2$ and $c' \leq 2^d/m$, let $\mathcal{P}_d^{c'}$ be the class of polynomials $\mathbb{F}_2^n \to \mathbb{F}_2$ of degree d that accept all but a $c' \cdot \left(m \cdot 2^{-d}\right)$ fraction of their inputs.*

Let d be an integer such that $\log(m) < d \leq \min\{\log(m) + \log(1/b(n)), n\}$, and let $2 < c' \leq 2^d/m$ be a real number. Assume that there exists a hitting-set generator G with density more than $(2/c') + m \cdot 2^{-d}$ for $\mathcal{P}_d^{c'}$. Then, G is a hitting-set generator for $\mathcal{C}$.

To obtain parameters as in Theorem 7, let $\epsilon = \epsilon(n)$ such that $2^{-n/2} \leq \epsilon \leq 1/8$, and let $m = m(n) \leq 2^{n/2}$. For $d = \lfloor \log(m) + \log(1/\epsilon) \rfloor \leq n$ and $c' = 4 \leq 2^d/m$, assume that there exists a hitting-set generator G for the class $\mathcal{P}_d^{c'}$ with density $1/2 + 2 \cdot \epsilon \geq (2/c') + m \cdot 2^{-d}$. Then, Proposition 36 asserts that G is a hitting-set generator for the class of $\oplus \wedge \oplus$ circuits with m $\wedge$-gates that accept all but $\epsilon \cdot 2^n$ of their inputs.

Proof. Let $C : \{0,1\}^n \to \{0,1\}$ be a $\oplus \wedge \oplus$ circuit with m $\wedge$-gates that accepts all but a $b(n)$ fraction of its inputs. We will show how to randomly compute C by a distribution that is typically in the class $\mathcal{P}_d^{c'}$, and then rely on Lemma 16 to deduce that any sufficiently dense hitting-set generator for $\mathcal{P}_d^{c'}$ also hits C.

The distribution over polynomials is obtained using Razborov's approximating polynomials method [17]. Our goal is to randomly replace each $\wedge$-gate g that has fan-in more than d with a polynomial $g' : \{0,1\}^n \to \{0,1\}$ of degree d such that for every fixed input $x \in \{0,1\}^n$ it holds that $g(x) = g'(x)$ with probability at least $1 - 2^{-d}$. To this purpose, given $g(x) = \wedge_{j=1}^{k} L_j(x)$, where $k > d$ and the L_j's are linear functions, we randomly choose d subsets $S_1, ..., S_d \subseteq [k]$, and replace g with the $\mathbb{F}_2$-polynomial $g'(x) = \Pi_{i=1}^{d} \left(1 + \sum_{j \in S_i} (L_i(x) + 1)\right)$. [18]

The above yields a random polynomial $p : \mathbb{F}_2^n \to \mathbb{F}_2$ of degree at most d such that for every fixed $x \in \{0,1\}^n$ it holds that $\Pr[p(x) = C(x)] \geq 1 - m \cdot 2^{-d}$. The expected fraction of unsatisfying inputs for p is at most $2m \cdot 2^{-d}$; this is because

$$\begin{aligned}
\mathbb{E}_p \left[\Pr_x [p(x) = 0] \right] &= \mathbb{E}_x \left[\Pr_p [p(x) = 0] \right] \\
&\leq \Pr_x[C(x) = 0] + \Pr_x[C(x) = 1] \cdot \max_x \left\{ \Pr_p[p(x) \neq C(x)] \right\} \\
&\leq b(n) + m \cdot 2^{-d} \,,
\end{aligned}$$

and since $d \leq \log(m) + \log(1/b(n))$ we have that $m \cdot 2^{-d} \geq b(n)$. Thus, the probability that the fraction of unsatisfying inputs for p is more than $c' \cdot \left(m \cdot 2^{-d}\right)$ is at most $2/c'$.

Thus, there exists a distribution that is $(1 - 2/c')$-typically in $\mathcal{P}_d^{c'}$ and that rejects every $x \notin C^{-1}(1)$ with probability at least $1 - m \cdot 2^{-d}$. Now, let $\mathbf{w}$ be the output distribution of a hitting-set generator with density $1 - c > (2/c') + m \cdot 2^{-d}$ for $\mathcal{P}_d^{c'}$. Relying on Lemma 16,

$$\Pr[C(\mathbf{w}) = 1] \geq 1 - m \cdot 2^{-d} - (2/c') - c > 0 \,,$$

which concludes the proof. ◀

[18] Using the standard analysis, if $g(x) = 1$, then $L_j(x) = 1$ for all $j \in [k]$, which implies that $g'(x) = 1$ with probability one; and if $g(x) = 0$, then for every $i \in [d]$, with probability $1/2$ over choice of S_i it holds that $\sum_{j \in S_i} (L_i(x) + 1) = 1$, which implies that $g'(x) = 0$ with probability $1 - 2^{-d}$.

7.2 Proof of Theorem 8

For this section, we first define and construct *multivalued OR functions*. We say that a function $f : \mathbb{F}^k \to \mathbb{F}$ is a multivalued OR function if $f(0, ..., 0) = 0$, and for every $x \neq (0, ..., 0)$ it holds that $f(x) \neq 0$. Indeed, for any non-zero input $x \neq (0, ..., 0)$, we require that f outputs *some* non-zero value.

▶ **Definition 37** (Multivalued OR Functions)**.** Let $\mathbb{F}$ be a finite field, and let k be an integer. We say that $f : \mathbb{F}^k \to \mathbb{F}$ is a multivalued OR function if for every $x \in \mathbb{F}^k$ such that $x \neq (0, 0, ..., 0)$ it holds that $f(x) \neq 0$.

Note that the function that outputs 1 on all non-zero inputs (and vanishes at $(0, ..., 0)$) satisfies Definition 37, but this function has a very high degree as a polynomial (i.e., it has degree $k \cdot |\mathbb{F} - 1|$, which is in fact the maximal degree). In contrast, we are interested in computing multivalued OR functions by polynomials of much lower degree. We now show that for any k, there exists a polynomial $\mathbb{F}^k \to \mathbb{F}$ of degree at most $2 \cdot k$ that computes a multivalued OR function of its k variables.

▶ **Proposition 38** (Construction of a Multivalued OR Function)**.** *Let $\mathbb{F}$ be a finite field, and let k be an integer. Then, there exists a polynomial $p : \mathbb{F}^k \to \mathbb{F}$ of degree $2^{\lceil \log(k) \rceil}$ that computes a multivalued OR function of its k variables.*

Proof. Let us first assume that k is a power of two. We want to construct a k-variate polynomial of degree k that vanishes only at $(0, ..., 0)$. We will first construct a bivariate polynomial that vanishes only at $(0, 0)$, and then recurse the construction, to repeatedly double the number of variables as well as the degree, while maintaining the invariant that the polynomial vanishes if and only if all of its inputs are zero.

Let $\alpha \in \mathbb{F}$ be a quadratic non-residue (i.e., for every $c \in \mathbb{F}$ it holds that $c^2 \neq \alpha$). The initial bivariate polynomial is defined by $f^{(2)}(x_1, x_2) = x_1^2 + \alpha \cdot x_2^2$. Observe that there does not exist a solution other than $(0, 0)$ to the equation $f^{(2)}(x_1, x_2) = 0$, since α is not a quadratic residue. Now, for every $k \geq 4$ that is a power of two, let $f^{(k)}(x_1, ..., x_k) = \left(f^{(k/2)}(x_1, ..., x_{k/2})\right)^2 + \alpha \cdot \left(f^{(k/2)}(x_{k/2+1}, ..., x_k)\right)^2$. Observe that $f^{(k)}(x_1, ..., x_k) = 0$ if and only if $x_i = 0$ for every $i \in [k]$, whereas $\deg(f^{(k)}) = k$. Finally, for any k that is not a power of two, we can use a straightforward padding argument to obtain a polynomial of degree $2^{\lceil \log(k) \rceil}$. ◀

We are now ready to prove the main claim that will be used in the proof of Theorem 8. The following proposition reduces the task of hitting any polynomial $p : \mathbb{F}^n \to \mathbb{F}$ of degree d to the task of hitting a polynomial $p' : \mathbb{F}^{t \cdot n} \to \mathbb{F}$ of degree $d' = \text{poly}(d)$ that vanishes very rarely.

▶ **Proposition 39** (Reducing Hitting Polynomials to Hitting Polynomials that Vanish Rarely)**.** *Let $t \geq 2$ be an even integer, and let $\epsilon > 0$ be a real number. Let $n \in \mathbb{N}$, let $\mathbb{F}$ be a finite field of cardinality $|\mathbb{F}| = q$, and let $1 \leq d \leq (1 - \epsilon) \cdot q$. Assume that there exists a hitting-set generator with seed length s for the class of polynomials $\mathbb{F}^{t \cdot n} \to \mathbb{F}$ of degree $d' = (2 \cdot d)^t$ that vanish on at most a $b(n) = O\left(q^{-t^2/4}\right)$ fraction of their inputs, where the O-notation hides a constant that depends on t and on ϵ. Then, there exists a hitting-set generator with seed length $s' = s + (t - 1) \cdot \lceil \log(q) \rceil$ for the class of all polynomials $\mathbb{F}^n \to \mathbb{F}$ of degree d.*

A high-level overview of the proof of Proposition 39 appeared in Section 2.4. We stress that the field size $|\mathbb{F}| = q$ is the same both for the polynomials $\mathbb{F}^n \to \mathbb{F}$ and for the polynomials $\mathbb{F}^{t \cdot n} \to \mathbb{F}$.

Proof. For any tuple of t elements $\vec{u} = \left(u^{(0)}, u^{(1)}, ..., u^{(t-1)}\right) \in \mathbb{F}^{t \cdot n}$, denote by $W_{\vec{u}} \subseteq \mathbb{F}^n$ the affine subspace $W_{\vec{u}} = \{u^{(0)} + \alpha_1 \cdot u^{(1)} + ... + \alpha_{t-1} \cdot u^{(t-1)} : \alpha_1, ..., \alpha_{t-1} \in \mathbb{F}\}$. Also, denote by $\mathcal{P}_{d'}$ the class of polynomials $\mathbb{F}^{t \cdot n} \to \mathbb{F}$ of degree d' that vanish on at most $b(n)$ of their inputs.

Our proof strategy is as follows. For any polynomial $p : \mathbb{F}^n \to \mathbb{F}$ of degree d, we will construct a corresponding polynomial $p' : \mathbb{F}^{t \cdot n} \to \mathbb{F}$ of degree at most $d' = (2 \cdot d)^t$ such that $p'(\vec{u}) = 0$ if and only if $p\restriction_{W_{\vec{u}}} \equiv 0$. We will show that with high probability over choice of $\vec{u}$ it holds that $p\restriction_{W_{\vec{u}}} \not\equiv 0$, which implies that the polynomial p' vanishes rarely; that is, we will show that $p' \in \mathcal{P}_{d'}$. Thus, for every $p : \mathbb{F}^n \to \mathbb{F}$ of degree d, a hitting-set generator G for $\mathcal{P}_{d'}$ also hits p', which means that the generator finds a subspace $W_{\vec{u}}$ such that $p\restriction_{W_{\vec{u}}} \not\equiv 0$. This allows us to find a satisfying input for p by invoking G and then choosing a random input in $W_{\vec{u}}$. Details follow.

Let us first fix an arbitrary $p : \mathbb{F}^n \to \mathbb{F}$, and construct the corresponding polynomial $p' : \mathbb{F}^{t \cdot n} \to \mathbb{F}$. For an input $\vec{u} \in \mathbb{F}^{t \cdot n}$ and $i \in [t]$, denote $u^{(i)} = (u_1^{(i)}, ..., u_n^{(i)}) \in \mathbb{F}^n$, and observe that the polynomial $p\restriction_{W_{\vec{u}}}(\alpha_1, ..., \alpha_{t-1})$ is of the form

$$\begin{aligned} p\restriction_{W_{\vec{u}}}(\alpha_1, ..., \alpha_{t-1}) &= p\left(u^{(0)} + \alpha_1 \cdot u^{(1)} + ... + \alpha_{t-1} \cdot u^{(t-1)}\right) \\ &= p\Big(u_1^{(0)} + \alpha_1 \cdot u_1^{(1)} + ... + \alpha_{t-1} \cdot u_1^{(t-1)}, ..., \\ &\qquad\qquad u_n^{(0)} + \alpha_1 \cdot u_n^{(1)} + ... + \alpha_{t-1} \cdot u_n^{(t-1)}\Big) \\ &= \sum_{i_1 + i_2 + ... + i_{t-1} \leq d} c_{i_1, ..., i_{t-1}}(\vec{u}) \cdot \alpha_1^{i_1} \cdot ... \cdot \alpha_{t-1}^{i_{t-1}} \,, \end{aligned} \tag{7.1}$$

where for every $i_1 + i_2 + ... + i_{t-1} \leq d$ it holds that $c_{i_1, ..., i_{t-1}}(\vec{u})$ is the coefficient of the monomial $\alpha_1^{i_1} \cdot ... \cdot \alpha_{t-1}^{i_{t-1}}$ in $p\restriction_{W_{\vec{u}}}$.

Note that $p\restriction_{W_{\vec{u}}} \equiv 0$ if and only if for every tuple $(i_1, ..., i_{t-1})$ such that $i_1 + ... + i_{t-1} \leq d$ it holds that $c_{i_1, ..., i_{t-1}}(\vec{u}) = 0$. Thus, we wish to construct a polynomial p' such that $p'(\vec{u}) \neq 0$ if and only if there exists $(i_1, ..., i_{t-1})$ such that $i_1 + ... + i_{t-1} \leq d$ and $c_{i_1, ..., i_{t-1}}(\vec{u}) \neq 0$. Note that the number of coefficients of $p\restriction_{W_{\vec{u}}}$ is $k = \binom{d+t-1}{t-1}$. The polynomial $p' : \mathbb{F}^{t \cdot n} \to \mathbb{F}$ is a multivalued OR function of these k coefficients $c_{i_1, ..., i_{t-1}}(\vec{u})$, which we construct using Proposition 38. To upper-bound the degree of p' (by d'), note that each $c_{i_1, ..., i_{t-1}}$ is a polynomial of degree at most d in $\vec{u}$.

▶ **Claim 39.1.** *For every $(i_1, ..., i_{t-1})$ such that $i_1 + ... + i_{t-1} \leq d$ it holds that $c_{i_1, ..., i_{t-1}}$, as defined in Eq. (7.1), is a polynomial of degree at most d in $\vec{u} = (u^{(0)}, ..., u^{(t-1)}) \in \mathbb{F}^{t \cdot n}$.*

Proof. Consider the polynomial $p\restriction_{W_{\vec{u}}}[\alpha_1, ..., \alpha_{t-1}]$ as a function of $\vec{u}$. By the definition of $p\restriction_{W_{\vec{u}}}$, it holds that $p\restriction_{W_{\vec{u}}}[\alpha_1, ..., \alpha_{t-1}] = p[\beta_1, ..., \beta_n]$, where for every $i \in [n]$ it holds that $\beta_i = u_i^{(0)} + \alpha_i \cdot u_i^{(1)} + ... + \alpha_{t-1} \cdot u_i^{(t-1)}$. Note that for every $i \in [n]$ it holds that β_i is a linear function of $\vec{u}$. Since p is of total degree d, the polynomial $p[\beta_1, ...\beta_n]$ is a sum of monomials of degree at most d in $\beta_1, ..., \beta_n$, and because each β_i is linear in $\vec{u}$, each such monomial is a polynomial of degree at most d in $\vec{u}$. ◀

Therefore, the degree of p' is less than $2 \cdot \binom{d+t-1}{t-1} \cdot d < (2 \cdot d)^t = d'$. Finally, let us upper-bound the probability that p' vanishes, in order to show that $p' \in \mathcal{P}_{d'}$. To do so, note that $\Pr_{x \in \mathbb{F}^n}[p(x) = 0] \leq d/q \leq 1 - \epsilon$ (where the first inequality is by the Schwartz-Zippel lemma, and the second inequality is by the hypothesis that $d \leq (1 - \epsilon) \cdot q$). Also recall that when uniformly choosing $\vec{u} \in \mathbb{F}^{t \cdot n}$, the points in $W_{\vec{u}}$ are t-wise independent. Relying on Fact 12, we deduce that:

▶ **Claim 39.2.** *The probability over choice of $\vec{u}$ that $p\restriction_{W_{\vec{u}}} \equiv 0$ is at most $O\left(d^{t/2} \cdot q^{-t^2/2}\right)$, where the O-notation hides a constant that depends on t and on ϵ.*

The proof of Claim 39.2 amounts to a straightforward calculation, so we defer it to Appendix C. Relying on Claim 39.2 and on the hypothesis that $d \leq (1-\epsilon) \cdot q$, we deduce that $\Pr_{\vec{u}}\left[p'(\vec{u}) = 0\right] = \Pr_{\vec{u}}\left[p\restriction_{W_{\vec{u}}} \equiv 0\right] < O\left(q^{-t^2/2+t/2}\right) \leq O\left(q^{-t^2/4}\right) = b(n)$.

Now, assuming that we have a hitting-set generator G for $\mathcal{P}_{d'}$, we construct a hitting-set generator for degree-d polynomials as follows. We invoke G to obtain a tuple $\vec{u} \in \mathbb{F}^{t \cdot n}$, and then use additional $(t-1) \cdot \lceil \log(q) \rceil$ bits of randomness to choose an element in the affine subspace $W_{\vec{u}}$. Since G finds $\vec{u}$ such that $p\restriction_{W_{\vec{u}}} \not\equiv 0$, with positive probability, our hitting-set generator hits p, with positive probability. ◀

Proposition 39 reduces the task of hitting a polynomial $\mathbb{F}^n \to \mathbb{F}$ of degree d to the task of hitting of a polynomial $p' : \mathbb{F}^{t \cdot n} \to \mathbb{F}$ of higher degree $d' = \text{poly}(d)$ that vanishes very rarely. The following proposition shows how to reduce the task of hitting p to the task of hitting polynomials of the *same degree* as p that vanish with probability at most $O(1/|\mathbb{F}|)$.

▶ **Proposition 40** (Reducing Hitting Polynomials to Hitting Polynomials of the Same Degree that Vanish Infrequently)**.** *Let $n \in \mathbb{N}$, and let $\mathbb{F}$ be a finite field of cardinality $|\mathbb{F}| = q$. For any $c' > 0$ and $d \geq 1$, let $\mathcal{P}_{d,c'}$ be the class of polynomials $\mathbb{F}^{2 \cdot n} \to \mathbb{F}$ of degree d that vanish on at most a $b(n) = c'/q$ fraction of their inputs. Then, for any integer d such that $d + 2\sqrt{d} \leq q$ and any $2 \leq c' \leq d$, the following holds:*

If there exists a hitting-set generator for the class $\mathcal{P}_{d,c'}$ with seed length $s = s(n, q, d, c')$ and density more than $2/c'$, then there exists a hitting-set generator for polynomials $\mathbb{F}^n \to \mathbb{F}$ of degree d with seed length $s' = s + \lceil \log(q) \rceil$.

Proof. The starting point of the current proof is the proof of Proposition 39, with the fixed parameter $t = 2$. [19] Let $G = \left\{\vec{u} \in \mathbb{F}^{2 \cdot n} : p\restriction_{W_{\vec{u}}} \not\equiv 0\right\}$ be the set of subspaces on which p is not identically zero. Our goal is to show a distribution $\mathbf{h}$ over polynomials $\mathbb{F}^{2 \cdot n} \to \mathbb{F}$ of degree d that satisfies the following:

- For every $\vec{u} \notin G$ it holds that $\Pr[\mathbf{h}(\vec{u}) = 0] = 1$.
- The probability that $h \sim \mathbf{h}$ vanishes on more than c'/q of its inputs is at most $2/c'$.

We can then rely on Lemma 16, to deduce that any sufficiently dense hitting-set generator for degree-d polynomials that vanish on at mos c'/q of their inputs also hits G, which allows us to hit p with additional $\lceil \log(q) \rceil$ random bits.

Towards constructing $\mathbf{h}$, recall that for every fixed $\vec{u} \in \mathbb{F}^{2 \cdot n}$, the $d+1$ coefficients of $p\restriction_{W_{\vec{u}}}$ are degree-d polynomials in $\vec{u}$, denoted $c_1(\vec{u}), ..., c_{d+1}(\vec{u})$. The distribution $\mathbf{h}$ is simply a random $\mathbb{F}$-linear combination of the c_i's. That is, for a random tuple $\vec{\beta} = (\beta_0, \beta_1, ..., \beta_d) \in \mathbb{F}^{(d+1) \cdot n}$, we define $h_{\vec{\beta}}(\vec{u}) = \sum_{i=0}^{d} \beta_i \cdot c_i(\vec{u})$. Note that for every $\vec{\beta} \in \mathbb{F}^{(d+1) \cdot n}$ it holds that $h_{\vec{\beta}}$ is of degree d. Also, if $\vec{u} \notin G$ (i.e., all the $c_i(\vec{u})$'s equal zero), then $h_{\vec{\beta}}(\vec{u}) = 0$ with probability one, and otherwise, $h_{\vec{\beta}}(\vec{u}) \neq 0$ with probability $1 - 1/q$.

We now show that at least a $(1 - 2/c')$ fraction of the $h_{\vec{\beta}}$'s vanish on at most c'/q of their inputs. Since the points in W are pairwise-independent, we have that:

[19] Larger values of t will not help to reduce the vanishing probability of the polynomials in the target of the reduction, due to the error of $1/q$ in the randomized computation of p'. However, larger values of t can help us relax the requirement that $d + 2\sqrt{d} \leq q$, and allow for slightly larger values of d (that are still below q). We do not pursue this direction in the current text.

▶ **Claim 40.1.** *For any $\epsilon > 0$, if $d \leq (1-\epsilon)\cdot q$, then the probability over choice of $\vec{u}$ that $p\restriction_{W_{\vec{u}}} \equiv 0$ is at most $4 \cdot \left(\frac{d}{\epsilon^2 \cdot q^2}\right)$.*

The proof of Claim 40.1 appears in Appendix C. In our case, we have that $d \leq (1-\epsilon)\cdot q$, where $\epsilon = \frac{2\sqrt{d}}{q}$ (because $d + 2\sqrt{d} \leq q$); therefore, Claim 40.1 implies that $\Pr_{\vec{u}}[\vec{u} \notin G] \leq 1/q$. Hence, over a random choice of $\vec{\beta}$, the expected fraction of inputs on which $h_{\vec{\beta}}$ vanishes is

$$\begin{aligned}\mathbb{E}_{\vec{\beta}}\left[\Pr_{\vec{u}}\left[h_{\vec{\beta}}(\vec{u}) = 0\right]\right] &= \mathbb{E}_{\vec{u}}\left[\Pr_{\vec{\beta}}\left[h_{\vec{\beta}}(\vec{u}) = 0\right]\right] \\ &\leq \Pr_{\vec{u}}[\vec{u} \notin G] + \Pr_{\vec{u}}[\vec{u} \in G] \cdot \max_{\vec{u}\in G}\left\{\Pr_{\vec{\beta}}[h_{\vec{\beta}}(\vec{u}) = 0]\right\},\end{aligned}$$

which is upper bounded by $2/q$. It follows that the probability that $h_{\vec{\beta}}$ vanishes on more than c'/q fraction of its inputs is at most $2/c'$.

Now, let $\mathbf{w}$ be the output distribution of a hitting-set generator with density $\mu > 2/c'$ for $\mathcal{P}_{d,c'}$; then, Lemma 16 implies that $\Pr[\mathbf{w} \in G] > \mu - 2/c' > 0$. Finally, similarly to the proof of Proposition 39, after obtaining $\vec{u} \in \mathbb{F}^{2\cdot n}$ we can use another $\log(q)$ bits to uniformly choose an element in $W_{\vec{u}}$, thus hitting p with positive probability. ◀

Let us now formally state Theorem 8, and prove it as a corollary of Propositions 39 and 40.

▶ **Theorem 41** (Theorem 8, Restated)**.** *Let $k \in \mathbb{N}$, let $t \geq 2$ be an even integer, and let $\epsilon > 0$ be a real number. Let $n \in \mathbb{N}$ be sufficiently large, and let $\mathbb{F}$ be a field of size $|\mathbb{F}| = q \leq n^k$. Then, the following holds:*

1. *Let $d \in \mathbb{N}$ such that $d \geq k+1$ and $d + 2\cdot\sqrt{d} \leq q$, and let $c' \in (2, d]$. Then, any hitting-set generator with density more than $2/c'$ for polynomials $\mathbb{F}^n \to \mathbb{F}$ of degree d that vanish on at most a $b(n) = c'/q$ fraction of their inputs requires seed of $\Omega\left(\log\left(\binom{n+d}{d}\right)\right)$ bits.*
2. *Let d' be an integer such that $(2k)^{t(t+1)} \leq d' \leq (1-\epsilon)\cdot q^{t+1}$. Then, any hitting-set generator for the class of polynomials $\mathbb{F}^n \to \mathbb{F}$ of degree d' that vanish on at most a $b(n) = O\left(q^{-t^2/4}\right)$ fraction of their inputs requires seed of $\Omega\left(\log\left(\binom{n+d}{d}\right)\right)$ bits, where $d = (d')^{1/(t+1)}$.*

In the two items above, the constants hidden in the Ω-notation of the lower bound may depend on k, on ϵ, and (in the first item) on t.

Proof. Recall that any hitting-set generator for the class of all polynomials $\mathbb{F}^n \to \mathbb{F}$ of degree d (i.e., without any assumption about their vanishing probability) must use a seed of at least $s' \geq \log\left(\binom{n+d}{d}\right)$ bits. This is the case because otherwise we can interpolate the $2^{s'} < \binom{n+d}{d}$ points in the image of the hitting-set generator by a non-zero degree-d polynomial. Also note that it suffices to prove the lower bounds for n that is a multiple of $t = O(1)$, due to a padding argument (i.e., because any hitting-set generator for polynomials $\mathbb{F}^n \to \mathbb{F}$ that vanish on at most $O\left(q^{-t^2/4}\right)$ of their inputs can be used as a hitting-set generator for polynomials $\mathbb{F}^{n-O(1)} \to \mathbb{F}$ that vanish on the same fraction of inputs, by adding dummy variables; and ditto for $O(1/q)$).

To prove Item (1), assume that there exists a hitting-set generator with seed length s and density more than $2/c'$ for polynomials of degree d that vanish on c'/q of their inputs. Relying on Proposition 40, there exists a hitting-set generator for all polynomials $\mathbb{F}^{n/2} \to \mathbb{F}$ of degree d with seed length $s' = s + \lceil \log q \rceil$. Since $s' \geq \log\left(\binom{n/2+d}{d}\right)$, we deduce that

$s \geq \log\left(\binom{n/2+d}{d}\right) - \lceil \log(q) \rceil = \Omega\left(\log\left(\binom{n/2+d}{d}\right)\right)$, where the equality holds because $q \leq n^k$ and $d \geq k+1$. Finally, we rely on the following elementary fact:

▶ **Fact 41.1.** *Let t be a constant integer. Let n and d be two integers such that the sum $n+d$ is sufficiently large. Then, we have that $\log\left(\binom{n/t+d}{d}\right) = \Omega\left(\log\left(\binom{n+d}{d}\right)\right)$, where the constant hidden inside the Ω-notation depends on t.*

The proof of Fact 41.1 appears in Appendix C. It follows from Fact 41.1 that $s \geq \Omega\left(\log\left(\binom{n+d}{d}\right)\right)$, which concludes the proof of Item (1).

The proof of Item (2) is similar to that of Item (1). Assume that there exists a hitting-set generator with seed length s for the class of degree-d' polynomials $\mathbb{F}^n \to \mathbb{F}$ that vanish on at most a $O\left(q^{-t^2/4}\right)$ fraction of their inputs. Let $d = \left\lfloor (d')^{1/t}/2 \right\rfloor$ (such that $d' \geq (2 \cdot d)^t$). According to Proposition 39, there exists a hitting-set generator for all polynomials $\mathbb{F}^{n/t} \to \mathbb{F}$ of degree d with seed length $s' = s + (t-1) \cdot \lceil \log(q) \rceil$. Since we know that $s' \geq \log\left(\binom{n/t+d}{d}\right)$, it holds that s is lower bounded by

$$\begin{aligned}
\log\left(\binom{n/t+d}{d}\right) - (t-1) \cdot \lceil \log(q) \rceil &= \Omega\left(\log\left(\binom{n/t+d}{d}\right)\right) \\
&= \Omega\left(\log\left(\binom{n+d}{d}\right)\right) \\
&= \Omega\left(\log\left(\binom{n+(d')^{1/(t+1)}}{(d')^{1/(t+1)}}\right)\right) ,
\end{aligned}$$

where the first equality is because $q \leq n^k$ and $d \geq \frac{(2k)^{t+1}}{2} \geq (t+1) \cdot k$, the second equality is due to Fact 41.1, and the last equality is because $d \geq (d')^{1/(t+1)}$. ◀

Acknowledgements. The author thanks his advisor, Oded Goldreich, for many helpful discussions, and for his guidance and support during the research and writing process. The author thanks Inbal Livni for very useful discussions about polynomials that vanish rarely, and Avishay Tal for very useful discussions about constant-depth circuits.

Part of this research was conducted during the workshop on small-depth circuits in St. Petersburg (May 2016), and the author is grateful to the organizers of the workshop.

References

1 M. Bellare and J. Rompel. Randomness-efficient oblivious sampling. In *Proc. 35th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 276–287, 1994.

2 Andrej Bogdanov. Pseudorandom generators for low degree polynomials. In *Proc. 37th Annual ACM Symposium on Theory of Computing (STOC)*, pages 21–30. 2005.

3 Andrej Bogdanov and Emanuele Viola. Pseudorandom bits for polynomials. *SIAM Journal of Computing*, 39(6):2464–2486, 2010.

4 Shiva Chaudhuri and Jaikumar Radhakrishnan. Deterministic restrictions in circuit complexity. In *Proc. 28th Annual ACM Symposium on Theory of Computing (STOC)*, pages 30–36, 1996.

5 Kuan Cheng and Xin Li. Randomness extraction in AC0 and with small locality. *Electronic Colloquium on Computational Complexity: ECCC*, 23:18, 2016.

6 Gil Cohen and Amnon Ta-Shma. Pseudorandom generators for low degree polynomials from algebraic geometry codes. *Electronic Colloquium on Computational Complexity: ECCC*, 20:155, 2013.

7 Anindya De, Omid Etesami, Luca Trevisan, and Madhur Tulsiani. Improved pseudorandom generators for depth 2 circuits. In *Proc. 14th International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM)*, pages 504–517, 2010.

8 Oded Goldreich. *Introduction to Property Testing (working draft)*, February 7, 2017. Accessed at `http://www.wisdom.weizmann.ac.il/~oded/pt-intro.html`, February 14, 2017.

9 Oded Goldreich and Avi Widgerson. On derandomizing algorithms that err extremely rarely. In *Proc. 46th Annual ACM Symposium on Theory of Computing (STOC)*, pages 109–118. 2014. Full version available online at *Electronic Colloquium on Computational Complexity: ECCC*, 20:152 (Rev. 2), 2013.

10 Parikshit Gopalan, Raghu Meka, and Omer Reingold. Dnf sparsification and a faster deterministic counting algorithm. *Computational Complexity*, 22(2):275–310, 2013.

11 Johan Håstad. *Computational Limitations of Small-depth Circuits*. MIT Press, 1987.

12 Russell Impagliazzo and Avi Wigderson. P = BPP if E requires exponential circuits: derandomizing the XOR lemma. In *Proc. 29th Annual ACM Symposium on Theory of Computing (STOC)*, pages 220–229. 1999.

13 Swastik Kopparty and Srikanth Srinivasan. Certifying polynomials for $\mathrm{AC}^0[\oplus]$ circuits, with applications. In *Proc. 32nd Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, pages 36–47. 2012.

14 Shachar Lovett, Omer Reingold, Luca Trevisan, and Salil Vadhan. Pseudorandom bit generators that fool modular sums. In *Proc. 13th International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM)*, pages 615–630. 2009.

15 Noam Nisan and Avi Wigderson. Hardness vs. randomness. *Journal of Computer and System Sciences*, 49(2):149–167, 1994.

16 Ryan O'Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2014.

17 Alexander A. Razborov. Lower bounds on the size of constant-depth networks over a complete basis with logical addition. *Mathematical Notes of the Academy of Science of the USSR*, 41(4):333–338, 1987.

18 Benjamin Rossman. The monotone complexity of k-clique on random graphs. *SIAM Journal of Computing*, 43(1):256–279, 2014.

19 Wolfgang M. Schmidt. *Equations over Finite Fields: An Elementary Approach*. Springer-Verlag Berlin, 1976.

20 Avishay Tal. Tight bounds on the fourier spectrum of $\mathcal{AC}^0$. *Electronic Colloquium on Computational Complexity: ECCC*, 21:174, 2014.

21 Roei Tell. Improved bounds for quantified derandomization of constant-depth circuits and polynomials. *Electronic Colloquium on Computational Complexity: ECCC*, 23, 2016. TR16-191. URL: `https://eccc.weizmann.ac.il/report/2016/191/`.

22 Luca Trevisan. Extractors and pseudorandom generators. *Journal of the ACM*, 48(4):860–879, 2001.

23 Luca Trevisan and TongKe Xue. A derandomized switching lemma and an improved derandomization of AC0. In *Proc. 28th Annual IEEE Conference on Computational Complexity (CCC)*, pages 242–247. 2013.

24 Salil P. Vadhan. *Pseudorandomness*. Foundations and Trends in Theoretical Computer Science. Now Publishers, 2012.

25 Emanuele Viola. The sum of d small-bias generators fools polynomials of degree d. *Computational Complexity*, 18(2):209–217, 2009.

A An alternative proof for Theorem 1.6 in [9]

Goldreich and Wigderson [9, Thm 1.6] proved that for any $d < n$, there exists a pseudorandom generator with seed length $O(\log(n))$ for the class of polynomials $p : \mathbb{F}_2^n \to \mathbb{F}_2$ of degree d that vanish at most a $b(n) = O\left(2^{-d}\right)$ fraction of their inputs (the theorem statement in [9] asserts the existence of a hitting-set generator, but in their proof they actually construct a pseudorandom generator). Their proof is based on a refinement of a lemma of Viola [25, Lemma 4]. We present an alternative proof of their result, which relies on Lemma 18.

High-level outline

Let $p : \mathbb{F}_2^n \to \mathbb{F}_2$ be a polynomial of degree d that vanishes on at most $b(n) = O\left(2^{-d}\right)$ of its inputs. We will randomly compute p by a distribution over polynomials of constant degree, and rely on Lemma 18 to deduce that any pseudorandom generator for polynomials of constant degree also "fools" p.

The family of polynomials of constant degree that we will use to randomly compute p is defined as follows. For $d' = d - O(1)$ and a tuple $\vec{r} = (r_1, ..., r_{d'}) \in \mathbb{F}_2^{d' \cdot n}$, let $h_{\vec{r}} : \mathbb{F}_2^n \to \mathbb{F}_2$ be defined by

$$h_{\vec{r}}(x) = 1 + \Delta_{\vec{r}} p(x) = 1 + \sum_{S \subseteq [d']} p\left(x + \sum_{i \in S} r_i\right) , \tag{A.1}$$

where $\Delta_{\vec{r}} p(x)$ is the iterated directional derivative of p in directions $r_1, ..., r_{d'}$ (for a definition see, e.g., [16, Def. 6.48]). Note that $h_{\vec{r}}$ is a polynomial of degree at most $d - d' = O(1)$. The family $\mathcal{H}$ of polynomials that we will use to randomly compute p is induced by all possible choices of $\vec{r} \in \mathbb{F}_2^{d' \cdot n}$; that is, $\mathcal{H} = \left\{h_{\vec{r}} : \vec{r} \in \mathbb{F}_2^{d' \cdot n}\right\}$.

The key argument is that for every fixed input $x \in \mathbb{F}_2^n$, when uniformly choosing $h_{\vec{r}} \in \mathcal{H}$, with sufficiently good probability it holds that $p(x) = h_{\vec{r}}(x)$. To see this, note that if for every non-empty $S \subseteq [d']$ it holds that $p\left(x + \sum_{i \in S} r_i\right) = 1$, then $\Delta_{\vec{r}} p(x) = p(x) + (2^{d'} - 1) = p(x) + 1$, which implies that $h_{\vec{r}}(x) = p(x)$. Since p vanishes on at most $b(n)$ of its inputs, the latter event happens with probability at least $1 - 2^{d'} \cdot b(n) = \Omega(1)$. Thus, relying on Lemma 18, any pseudorandom generator for $\mathcal{H}$ also "fools" p. Let us now formalize and parametrize this argument.

▶ **Theorem 42** ($\mathbb{F}_2$-Polynomials with $b(n) = O(2^{-d})$)**.** *Let $c > 0$ be an arbitrarily large constant. Let $n \in \mathbb{N}$, let $d < n$, and let $p : \mathbb{F}_2^n \to \mathbb{F}_2$ be a polynomial of degree d that vanishes on at most $b(n) = c \cdot \left(2^{-d}\right)$ of its inputs. Then, for every $\delta > 0$, any pseudorandom generator with error $\delta/2$ for polynomials of degree $\lceil \log(2c/\delta) \rceil$ is also a pseudorandom generator with error δ for p, where pseudorandom generators for $\mathbb{F}_2$-polynomials are defined in Definition 9.*

Proof. Let $d' = d - \lfloor \log(2c/\delta) \rfloor$, let $\mathcal{H} = \left\{h_{\vec{r}} : \vec{r} \in \mathbb{F}_2^{d' \cdot n}\right\}$ such that for every $\vec{r} \in \mathbb{F}_2^{d' \cdot n}$ the function $h_{\vec{r}}$ is defined as in Eq. (A.1), and let $\mathbf{h}$ be the uniform distribution over $\mathcal{H}$. Note that for every fixed $x \in \mathbb{F}_2^n$ it holds that $\Pr[\mathbf{h}(x) = p(x)] > 1 - \delta/2$; this is the case because for every non-empty $S \subseteq [d']$, the probability that $p(x + \sum_{i \in S} r_i) = 0$ is at most $b(n)$, which implies that with probability at least $1 - b(n) \cdot (2^{d'} - 1) > 1 - \frac{\delta}{2}$ we have that $\mathbf{h}(x) = 1 + p(x) + \left(2^{d'} - 1\right) = p(x)$.

Now, let $\xi : \mathbb{F}_2 \to \mathbb{C}$ be the character $\xi(x) = (-1)^x$. Let $\mathbf{w}$ be a distribution that $(\delta/2)$-fools polynomials of degree $\lceil \log(2c/\delta) \rceil$ (which implies that for every such polynomial p' it holds that $\left|\mathbb{E}[\xi(p'(\mathbf{w}))] - \mathbb{E}[\xi(p'(\mathbf{u}_n))]\right| \leq \delta$). According to Lemma 18, using the parameter

values $\delta = \max_{x \in \mathbb{F}_2}\{|\xi(x)|\} = 1$, and $\epsilon_1 = (\delta/2)$, and $\epsilon_2 = 0$, and $\epsilon_3 = \delta$, it holds that $\left|\Pr[p(\mathbf{w}) = 1] - \Pr[p(\mathbf{u}_n) = 1]\right| = \frac{1}{2} \cdot \left|\mathbb{E}[\xi(p(\mathbf{w}))] - \mathbb{E}[\xi(p(\mathbf{u}_n))]\right| \leq \delta$. ◀

B Proofs of claims from Section 5

We prove two claims from Section 5.2.2 (i.e., Lemma 27 and a generalization of the switching lemma of [9]) and a technical claim from Section 5.2.1 (i.e., Claim 23). Lemma 27 is an adaptation of the main lemma of Trevisan and Xue [23]. Let us now recall the statement of Lemma 27, and prove the lemma.

▶ **Lemma 43** (Lemma 27, Restated). *Let F be a CNF over n inputs with m clauses, each clause of width at most w. For a positive parameter $p = 2^{-q}$, where $q \in \mathbb{N}$, let $\rho \in \{0,1,\star\}^n$ be a restriction that is chosen according to a distribution over $\{0,1\}^{(q+1)\cdot n}$ that δ_0-fools all CNFs of width $w' = w \cdot (q+1)$. Then, the probability that $F\restriction_\rho$ cannot be computed by a decision tree of depth D is at most $2^{D+w+1} \cdot (5pw)^D + \delta_0 \cdot 2^{(D+1)\cdot(2\cdot w + \log(m))}$.*

Proof Sketch. We rely on the proof of Lemma 7 in [23], and in particular use the same definitions of canonical decision tree, path, and segment. The proof in [23] reduces the task of finding a restriction ρ such that $F\restriction_\rho$ can be computed by a shallow decision tree to the task of "fooling" less than $2^{(D+1)\cdot(2w+\log(m))}$ tests: For each path of length $D+1$ (i.e., a sequence of $D+1$ segments), there is a corresponding test $T_P : \{0,1\}^{(q+1)\cdot n} \to \{0,1\}$ that gets as input a restriction $\rho \in \{0,1\}^{(q+1)\cdot n}$, and accepts ρ if and only if the canonical decision tree for $F\restriction_\rho$ contains the path P. Indeed, if all the tests reject ρ, it means that no path of length $D+1$ exists in the canonical decision tree for $F\restriction_\rho$, which implies that the canonical decision tree for $F\restriction_\rho$ is of depth D.

The key claim in the proof is Claim 8, which asserts that for each path P, the test T_P can be computed by a CNF. The goal in [23] is to show that the CNF for T_P has few clauses; we focus on showing that the CNF for T_P has small width. To see that this holds, note that T_P is constructed as a conjunction of conditions, where each condition depends only on the assignment that ρ gives to the variables of a single clause of F (either a clause that belongs to a segment in the path, or a clause whose index is between the indices of clauses that belong to segments in the path). Thus, each condition depends only on the assignment that ρ gives to w variables, which means that each condition depends only on $w' = w \cdot (q+1)$ bits of ρ. Hence, each condition can be decided by a CNF of width w', and T_P (which is their conjunction) can also be decided by a CNF of width w'. ◀

Let us now formally state the generalization of the switching lemma of Goldreich and Wigderson [9] and prove it.

▶ **Proposition 44** (A Generalization of the Derandomized Switching Lemma of [9]). *Let $m : \mathbb{N} \to \mathbb{N}$, let $w : \mathbb{N} \to \mathbb{N}$, and let $\delta : \mathbb{N} \to [0,1)$. Let $\boldsymbol{z}$ be a distribution over $\{0,1\}^{O(\log(w))\cdot n}$ that is δ'-almost t'-wise independent, where $\log(1/\delta') = O(t') = \tilde{O}(w) \cdot 2^w \cdot \log(1/\delta)$.*

Then, for any depth-2 formula $F : \{0,1\}^n \to \{0,1\}$ of width $w = w(n)$ with $m = m(n)$ clauses, with probability at least $1 - 4\delta$ (where $\delta = \delta(n)$) over choice of $\rho \sim \boldsymbol{z}$ it holds that the restricted formula $F\restriction_\rho$ can be computed by a decision tree of depth $D = O(\log(1/\delta))$.

Proof. Let $\delta_0 = \delta \cdot 2^{-D} = \mathrm{poly}(\delta)$, and fix a depth-2 formula $F : \{0,1\}^n \to \{0,1\}$; without loss of generality, assume that F is a CNF. [20] Consider a uniformly-chosen restriction ρ that

[20] This is without loss of generality since if F is a DNF, then $F\restriction_\rho$ can be computed by a depth-D decision tree if and only if $(\neg F)\restriction_\rho$ can be computed by such a tree.

keeps each variable alive with probability $p = 1/O(w)$; Hastad's switching lemma asserts that with probability at least $1 - 2^{-O(D)} \geq 1 - \delta_0$, the *canonical decision tree* of $F\restriction_\rho$ is of depth $D = O(\log(1/\delta))$ (the canonical decision tree is the decision tree that is constructed by the algorithm in Hastad's original proof; for a definition see, e.g., [23, Def. 4]).

Given a restriction ρ, we consider the following way to decide whether the canonical decision tree of $F\restriction_\rho$ is of depth D. Associate each string $P \in \{0,1\}^D$ with a potential *positional path* of depth D in the canonical decision tree of F; that is, the string P induces a path from the root to a specific node of depth D in a full binary tree of depth D or more. For each $P \in \{0,1\}^D$, we consider a corresponding test T_P that gets ρ as input, and tests whether or not one of the nodes in the path induced by P along the canonical decision tree of $F\restriction_\rho$ is a leaf node (i.e., whether or not the path ends at depth at most D); if there is indeed a leaf then T_P accepts ρ, and otherwise (i.e., if the path continues to depth $D+1$) then T_P rejects ρ. We will describe T_P in detail in a moment, but for now observe that the canonical decision tree of $F\restriction_\rho$ is of depth D if and only if for each $P \in \{0,1\}^D$ it holds that $T_P(\rho) = 1$.

To describe how each T_P works, fix $P \in \{0,1\}^D$, and let T_P be the following recursive algorithm. The algorithm gets as input a CNF F', a restriction ρ' and a string P' (in the first recursive call $F' = F$, $\rho' = \rho$, and $P' = P$). If the CNF is empty (i.e., has no clauses), then the algorithm accepts; otherwise, the algorithm examines the values that ρ' assigns to the variables in the first clause of F':

- If the first clause is unsatisfied by ρ' (i.e., all variables are fixed to unsatisfying values) then the algorithm accepts and halts.
- If the first clause is satisfied by ρ' (i.e., one or more variables are assigned to satisfying values), then the algorithm simplifies F' by omitting the first clause, and by simplifying the other clauses according to the values that ρ' assigned to the variables in the first clause. Then, the algorithm recurses with with the simplified CNF and with the same restriction ρ' and string P'.
- Otherwise, the first clause is undetermined by ρ'. If the number of living variables in the clause, denoted by k, is greater than the length of P', then the algorithm rejects. [21] If $k \leq |P'|$, let ρ'' be the restriction that fixes the k variables to values according to the k-prefix of P'. The algorithm simplifies F' according to the composition $\rho'' \circ \rho'$, and recurses with the simplified CNF, with the restriction $\rho'' \circ \rho'$, and with the string obtained from P' by omitting its first k bits.

The main point to note in the above description is that in each recursive call, the test T_P needs to read at most w blocks of $\lceil \log(1/p) \rceil = O(\log(w))$ bits in the restriction, corresponding to the (at most w) variables in the clause that it examines. The key observation in [9, Lemma 3.3], which we now state in a more general form, is that for each $P \in \{0,1\}^D$, with high probability it holds that T_P makes at most $D' = O\left(2^w \cdot \log\left(1/\delta_0\right)\right))$ recursive calls; that is, with high probability T_P examines the values that ρ assigns to variables of at most D' clauses. This is the case because for each recursive call, the probability that the clause that is examined is *unsatisfied* is at least 2^{-w}; thus, the probability that after D' recursive calls the algorithm encountered an unsatisfied clause, and thus stopped, is more than $1 - (1 - 2^{-w})^{D'} \geq 1 - \delta_0$. It follows that for each $P \in \{0,1\}^D$, with probability at least

[21] This event means that the path induced by P in the canonical decision tree of $F\restriction_\rho$ is of depth more than $|P| = D$. Recall that by the definition of the canonical decision tree, whenever the algorithm that constructs the canonical decision tree encounters an undetermined clause, it adds the full sub-tree that corresponds to all living variables in the clause to the canonical decision tree.

$1-2\delta_0$ over a uniformly-chosen restriction ρ it holds that T_P accepts ρ without making more than D' recursive calls.

Now, consider "truncated" versions of these tests: For each $P \in \{0,1\}^D$, consider a modified version T'_P of T_P that, in addition to the description above, rejects ρ if the depth of the recursion exceeds D'. According to previous paragraph, the test T'_P accepts a uniformly-chosen restriction with probability at least $1-2\delta_0$. Since each T'_P reads at most $D'' = O\left(D' \cdot w \cdot \log(w)\right) = \tilde{O}\left(w\right) \cdot \left(2^w \cdot \log(1/\delta)\right)$ bits in the restriction, if instead of the uniform distribution we choose a restriction from the distribution $\mathbf{z}$, which is δ'-almost t'-wise independent, where $\delta' < \left(\delta_0 \cdot 2^{-D''}\right)$ and $t' \geq D''$, then the probability that T'_P will accept is at least $1-3\delta_0$. [22] Thus, the probability that all the tests accept (i.e., $\wedge_{P\in\{0,1\}^D} T_P(\rho) = 1$) is at least $1-3\delta$. ◀

Let us now recall the statement of Claim 23 and prove it.

▶ **Claim 45** (Claim 23, Restated). *Let $F : \{0,1\}^n \to \{0,1\}$ be a depth-2 formula of width w and size m, and let $F' : \{0,1\}^n \to \{0,1\}$ be a refinement of F. Then, for any restriction $\rho \in \{0,1,\star\}^n$ it holds that $F{\restriction_\rho}$ can be computed by a depth-2 formula Φ of width w and size m such that $F'{\restriction_\rho}$ is a refinement of Φ.*

Proof. We prove the claim for the case where F is a DNF; the proof for the case where F is a CNF follows by reduction to the DNF $\neg F$, relying on Fact 21. Let Φ be the DNF for $F{\restriction_\rho}$ that is obtained by fixing the variables in each clause of F according to ρ, without omitting any clause from the formula (even if a clause becomes a constant function).

When F' was obtained by a sequence of removal steps and clean-up steps, then F' is simply a sub-formula of F. In this case, we can apply the same sequence of removal steps and clean-up steps to Φ, to obtain a corresponding sub-formula of Φ that computes $F'{\restriction_\rho}$. [23] We thus focus on proving the claim when F' was obtained by a sequence of $k \leq m$ merging steps and clean-up steps.

For every $i \in [k]$, let $F^{(i)}$ be the formula in the beginning of the i^{th} refinement step in the transformation of F to F', and let $F^{(k+1)} = F'$. We will show a sequence of k merging steps and clean-up steps that, when applied to Φ, induce a corresponding sequence of formulas $\Phi = \Phi^{(1)}, ..., \Phi^{(k+1)}$, such that the following holds: For every $i \in [k]$ there exists a bijection between the clauses of $\Phi^{(i)}$ and the clauses of $F^{(i)}{\restriction_\rho}$ such that every clause φ of the former is mapped to a clause f of the latter such that φ computes the function $f{\restriction_\rho}$. In particular, this claim implies that for every $i \in [k]$ it holds that $\Phi^{(i)} \equiv F^{(i)}{\restriction_\rho}$, and therefore $F'{\restriction_\rho} \equiv \Phi^{(k+1)}$ is a refinement of $\Phi = \Phi^{(1)}$.

The claim is proved by induction on i. The base case $i = 1$ follows immediately from the definition of $\Phi^{(1)} = \Phi$. For the induction step, assume that there is a bijection as above between the clauses of $\Phi^{(i)}$ and the clauses of $F^{(i)}{\restriction_\rho}$, and let us define the i^{th} refinement step that is applied to $\Phi^{(i)}$. If the i^{th} refinement step of $F^{(i)}$ was a clean-up step, then we

[22] The reason that we use the error parameter $\delta_0 \cdot 2^{-D''}$ instead of the more natural parameter δ_0 is that the tests that we are trying to "fool" are *adaptive*; that is, for each $P \in \{0,1\}^D$, the test T_P does not examine a fixed set of D'' bits in ρ, but rather adaptively chooses which bits to read according to the values of the bits that it read so far. We rely on the fact that any distribution that is $\left(\delta_0 \cdot 2^{-D''}\right)$-almost D''-wise independent also δ_0-fools adaptive tests that only read D'' bits (see, e.g., [8, Exer. 7.4]).

[23] That is, let $F = \vee_{i=1}^m f_i$, and assume that $F' = \vee_{i=k+1}^m f_i$ was obtained from F by removing the clauses $f_1, ..., f_k$. Then it holds that $\Phi = \vee_{i=1}^m (f_i{\restriction_\rho})$ and $F'{\restriction_\rho} = \vee_{i=k+1}^m (f_i{\restriction_\rho})$, which implies that we can apply k removal steps to Φ in order to obtain $F'{\restriction_\rho}$.

can apply an analogous clean-up step to $\Phi^{(i)}$. [24] Otherwise, if the i^{th} refinement step of $F^{(i)}$ was a merging step, let $f_1^{(i)},...,f_u^{(i)}$ be the set of clauses that were removed in this step, and let $h^{(i)}$ be the new clause that was added in their stead. For every $j \in [u]$, let $\varphi_j^{(i)}$ be the clause in $\Phi^{(i)}$ that computes $f_j^{(i)}\restriction_\rho$ and exists by the induction hypothesis. We show how apply a single refinement step to $\Phi^{(i)}$ that replaces the clauses $\varphi_1^{(i)},...,\varphi_u^{(i)}$ with a new clause $\varphi^{(i)}$ that computes the function $h^{(i)}\restriction_\rho$. This is proved by a case analysis:

1. If $h^{(i)}\restriction_\rho$ is not a constant function, then it follows that $\bigcap_{j\in[u]}(f_j^{(i)}\restriction_\rho) = \bigcap_{j\in[u]}\varphi_j^{(i)} \neq \emptyset$. In this case, we apply a merging step to the clauses $\varphi_1^{(i)},...,\varphi_u^{(i)}$ in $\Phi^{(i)}$, and they are replaced with the non-constant clause $\varphi^{(i)} = \bigcap_{j\in[u]}\varphi_j^{(i)} = \bigcap_{j\in[u]}(f_j^{(i)}\restriction_\rho) = h^{(i)}\restriction_\rho$.
2. If $h^{(i)}\restriction_\rho \equiv 0$, then for every $j \in [u]$ it holds that $f_j^{(i)}\restriction_\rho \equiv \varphi_j^{(i)} \equiv 0$. This is the case because $\bigcap_{j\in[u]} f_j^{(i)} \neq \emptyset$ (otherwise $h^{(i)} \equiv 1$ and also $h^{(i)}\restriction_\rho \equiv 1$), whereas $\left(\bigcap_{j\in[u]} f_j^{(i)}\right)\restriction_\rho \equiv 0$, which implies that for every $j \in [u]$ there exists a literal in $f_j^{(i)}$ that is fixed by ρ to an unsatisfying value. Therefore, in this case we can apply a clean-up step to $\Phi^{(i)}$ to remove all but a single constant zero clause among the $f_j^{(i)}$'s.
3. If $h^{(i)}\restriction_\rho \equiv 1$, then it holds that $\bigcap_{j\in[u]}\varphi_j^{(i)} = \emptyset$. To see that this is the case, note that if $\bigcap_{j\in[u]} f_j^{(i)} = \emptyset$ then the latter assertion holds immediately; and otherwise (i.e., $\bigcap_{j\in[u]} f_j^{(i)} \neq \emptyset$), it follows by the assumption that $h^{(i)}\restriction_\rho \equiv 1$ that ρ fixes all the literals that are shared by all the u clauses $f_1^{(i)},...,f_u^{(i)}$ to satisfying values, which indeed implies that $\bigcap_{j\in[u]}\varphi_j^{(i)} = \emptyset$. Thus, we can apply a merging step to $\varphi_1^{(i)},...,\varphi_u^{(i)}$ to obtain the constant one function. ◀

C Proofs of technical claims from Section 7

In this appendix we prove several technical claims that were made in the proofs of Proposition 39, Proposition 40, and Theorem 41.

Let us first prove a claim that generalizes Claims 39.2 and 40.1, which were made in the proofs of Proposition 39 and Proposition 40, respectively. Recall that for any tuple of t elements $\vec{u} = (u^{(0)},...,u^{(t-1)}) \in \mathbb{F}^{t\cdot n}$, we denote by $W_{\vec{u}} \subseteq \mathbb{F}^n$ the affine subspace $W_{\vec{u}} = \{u^{(0)} + \alpha_1\cdot u^{(1)} + ... + \alpha_{t-1}\cdot u^{(t-1)} : \alpha_1,...,\alpha_{t-1} \in \mathbb{F}\}$. Then, the following holds:

▶ **Claim 46** (Claims 39.2 and 40.1, Generalized). *Let $t \geq 2$ be an even integer, and let $\epsilon \in (0,1)$. Let $n \in \mathbb{N}$, let $\mathbb{F}$ be a field of size $|\mathbb{F}| = q$, and let $p : \mathbb{F}^n \to \mathbb{F}$ be a polynomial of degree $d \leq (1-\epsilon)\cdot q$. Uniformly choose $\vec{u} = (u^{(0)},...,u^{(t-1)}) \in \mathbb{F}^{t\cdot n}$, and let $W = W_{\vec{u}}$. Then, the probability that $p\restriction_W \equiv 0$ is at most $O\left(d^{t/2}\cdot q^{-t^2/2}\cdot\epsilon^{-t}\right)$, where the O-notation hides a constant that depends on t; in particular, when $t = 2$, the hidden constant is just 4.*

Proof. For $i = 1,...,q^{t-1}$, let $\mu_W^{(i)}$ be the indicator variable of whether p vanishes on the i^{th} point in W (according to some canonical ordering of points in $\mathbb{F}^n$), and let $\mu_W = \mathbb{E}_{i\in[q^{t-1}]}\left[\mu_W^{(i)}\right] = \Pr_{\vec{x}\in W}[p(\vec{x}) = 0]$. Denote by $b = \Pr_{x\in\mathbb{F}^n}[p(x) = 0]$, and note that

[24] Specifically, denote by $f_1^{(i)},...,f_u^{(i)}$ the constant zero clauses that were removed from $F^{(i)}$ in the i^{th} step. For every $j \in [u]$, let $\varphi_j^{(i)}$ be the clause in $\Phi^{(i)}$ that computes $f_j^{(i)}\restriction_\rho \equiv 0$ and exists by the induction hypothesis. Then, the i^{th} refinement step of $\Phi^{(i)}$ is a clean-up step that removes the constant zero clauses $\varphi_1^{(i)},...,\varphi_u^{(i)}$.

$b \leq d/q \leq 1-\epsilon$, where the first inequality is by the Schwartz-Zippel lemma, and the second inequality is by the hypothesis that $d \leq (1-\epsilon)\cdot q$.

We handle the case of $t=2$ and the case of $t \geq 4$ separately. Starting with the former, note that for every $i \neq j \in [q]$ it holds that $\mu_W^{(i)}$ and $\mu_W^{(j)}$ are independent, and that $Var\left(\mu_W^{(i)}\right) \leq b$. Relying on Chebyshev's inequality, we have that

$$\Pr_W[|\mu_W - b| > \epsilon/2] \leq \frac{b}{(\epsilon/2)^2 \cdot q} \leq 4 \cdot \left(\frac{d}{\epsilon^2 \cdot q^2}\right) .$$

For the case of $t \geq 4$, we rely on Fact 12. In our case, the t-wise independent variables are $\mu_W^{(1)}, ..., \mu_W^{(q^{t-1})}$, their average is $\frac{1}{q^{t-1}} \cdot \sum_{i \in [q^{t-1}]} \mu_W^{(i)} = \mu_W$, and their expected average is $b \leq 1-\epsilon$. Using Fact 12 with $\zeta = \epsilon/2$, we have that

$$\begin{aligned}
\Pr_W[|\mu_W - b| \geq \epsilon/2] &\leq 8 \cdot \left(\frac{t \cdot b \cdot q^{t-1} + t^2}{(\epsilon/2)^2 \cdot (q^{t-1})^2}\right)^{t/2} \\
&\leq 8 \cdot \left(\frac{2 \cdot t^2 \cdot \max\left\{b, q^{-(t-1)}\right\}}{(\epsilon/2)^2 \cdot q^{t-1}}\right)^{t/2} \\
&\leq \left(8 \cdot 2^{t/2} \cdot (2t)^t\right) \cdot \left(\frac{d/q}{\epsilon^2 \cdot q^{t-1}}\right)^{t/2} ,
\end{aligned}$$

which is $O\left(d^{t/2} \cdot q^{-t^2/2} \cdot \epsilon^{-t}\right)$. ◀

We now prove Fact 41.1, which was stated in the proof of of Theorem 41:

▶ **Fact 47** (Fact 41.1, Restated). *Let t be a constant integer. Let n and d be two integers such that the sum $n+d$ is sufficiently large. Then, we have that* $\log\left(\binom{n/t+d}{d}\right) = \Omega\left(\log\left(\binom{n+d}{d}\right)\right)$, *where the constant hidden inside the Ω-notation depends on t.*

Proof. Let $c = \frac{1}{t \cdot e}$, where $e = 2.71...$. If $d \leq c \cdot (n/t+d)$, then the assertion follows from the standard bound $\left(\frac{n}{k}\right)^k \leq \binom{n}{k} \leq \left(\frac{n \cdot e}{k}\right)^k$. [25] Similarly, if $(n/t) \leq c' \cdot (n/t+d)$, where $c' = 1/e$, then the assertion follows by showing that $\log\left(\binom{n/t+d}{n/t}\right) = \Omega\left(\log\left(\binom{n+d}{n}\right)\right)$, relying on the same standard bound. [26]

Otherwise, we have that $d > c \cdot (n/t+d)$ and $n/t > c' \cdot (n/t+d)$. In this case we use Stirling's approximation: Let $H_2(\cdot)$ be the binary entropy function, and denote $\alpha = \frac{d}{d+n}$ and $\alpha' = \frac{d}{d+(n/t)}$. Note that $\frac{c}{t} < \alpha < 1-c'$, and that $c < \alpha' < 1-c'$, which implies that $H_2(\alpha) = \Omega(1)$ and $H_2(\alpha') = \Omega(1)$. Hence, we deduce that $\log\left(\binom{n+d}{d}\right) \leq H_2(\alpha) \cdot (n+d)$, whereas $\log\left(\binom{n/t+d}{d}\right) \geq (H_2(\alpha') - o(1)) \cdot (n/t+d) = \Omega\left(H_2(\alpha) \cdot (n+d)\right)$. ◀

[25] Specifically, $\log\left(\binom{n+d}{d}\right) \leq d \cdot \left(\log\left(\frac{n+d}{d}\right) + \log(e)\right) < d \cdot \left(\log\left(\frac{(n/t)+d}{d}\right) + \log(t \cdot e)\right) \leq 2 \cdot d \cdot \log\left(\frac{(n/t)+d}{d}\right) \leq 2 \cdot \log\left(\binom{n/t+d}{d}\right)$, where the penultimate inequality relies on the fact that $\frac{(n/t)+d}{d} \geq t \cdot e$.

[26] Specifically, $\log\left(\binom{n+d}{n}\right) \leq n \cdot \left(\log\left(\frac{n+d}{n}\right) + \log(e)\right) < n \cdot \left(\log\left(\frac{(n/t)+d}{(n/t)}\right) + \log(e)\right) \leq 2 \cdot n \cdot \log\left(\frac{(n/t)+d}{(n/t)}\right) \leq (2 \cdot t) \cdot \log\left(\binom{n/t+d}{n/t}\right)$, where the penultimate inequality relies on the fact that $\frac{(n/t)+d}{n/t} \geq e$.

Bounded Independence Plus Noise Fools Products*

Elad Haramaty[1], Chin Ho Lee[2], and Emanuele Viola[3]

1 **Harvard John A. Paulson School of Engineering and Applied Sciences, Cambridge, MA, USA**
2 **College of Computer and Information Science, Northeastern University, Boston, MA, USA**
3 **College of Computer and Information Science, Northeastern University, Boston, MA, USA**

Abstract

Let D be a b-wise independent distribution over $\{0,1\}^m$. Let E be the "noise" distribution over $\{0,1\}^m$ where the bits are independent and each bit is 1 with probability $\eta/2$. We study which tests $f\colon \{0,1\}^m \to [-1,1]$ are ε-fooled by $D+E$, i.e., $|\operatorname{E}[f(D+E)] - \operatorname{E}[f(U)]| \leq \varepsilon$ where U is the uniform distribution.

We show that $D+E$ ε-fools product tests $f : (\{0,1\}^n)^k \to [-1,1]$ given by the product of k bounded functions on disjoint n-bit inputs with error $\varepsilon = k(1-\eta)^{\Omega(b^2/m)}$, where $m = nk$ and $b \geq n$. This bound is tight when $b = \Omega(m)$ and $\eta \geq (\log k)/m$. For $b \geq m^{2/3}\log m$ and any constant η the distribution $D+E$ also 0.1-fools log-space algorithms.

We develop two applications of this type of results. First, we prove communication lower bounds for decoding noisy codewords of length m split among k parties. For Reed–Solomon codes of dimension m/k where $k = O(1)$, communication $\Omega(\eta m) - O(\log m)$ is required to decode one message symbol from a codeword with ηm errors, and communication $O(\eta m \log m)$ suffices. Second, we obtain pseudorandom generators. We can ε-fool product tests $f\colon (\{0,1\}^n)^k \to [-1,1]$ under any permutation of the bits with seed lengths $2n + \tilde{O}(k^2 \log(1/\varepsilon))$ and $O(n) + \tilde{O}(\sqrt{nk}\log 1/\varepsilon)$. Previous generators have seed lengths $\geq nk/2$ or $\geq n\sqrt{nk}$. For the special case where the k bounded functions have range $\{0,1\}$ the previous generators have seed length $\geq (n + \log k)\log(1/\varepsilon)$.

1998 ACM Subject Classification F.0 Theory of Computation, E.4 Coding and Information Theory, G.3 Probability and Statistics

Keywords and phrases Bounded independence, Noise, Product tests, Error-correcting codes, Pseudorandomness

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.14

1 Introduction and our results

At least since the seminal work [17] the study of bounded independence has received a lot of attention in theoretical computer science. In particular, researchers have analyzed various classes of tests that cannot distinguish distributions with bounded independence from uniform. Such tests include (combinatorial) rectangles [25] (cf. [18]), bounded-depth

* Supported by NSF grant CCF-1319206. Work done in part while at the College of Computer and Information Science, Northeastern University, in part while visiting Harvard University, with support from Salil Vadhan's Simons Investigator grant, and in part while at the Simons Institute for the Theory of Computing.

COMPUTATIONAL COMPLEXITY CONFERENCE

circuits [11, 44, 16, 49], and halfspaces [22, 30, 23], to name a few. We say that such tests are *fooled* by distributions with bounded independence.

In this work we consider fooling tests which are a product of several functions on disjoint inputs, and hence are called *product tests*.

▶ **Definition 1** (Product Tests)**.** A *product test* with k functions of input length n and alphabet size s is a function $f\colon ([s]^n)^k \to \mathbb{C}_1$ which is the product of k functions $f_1, f_2, \ldots, f_k$ on disjoint inputs, where each f_i maps $[s]^n$ to $\mathbb{C}_1$, the complex unit disk $\{z \in \mathbb{C} : |z| \leq 1\}$.

We note that these tests make sense already for $n = 1$ and large s (and in fact as we will see have been considered for such parameters in the literature). But it is essential for our applications that the input set of the f_i has a product structure, so we think of n being large. We can choose $s = 2$ for almost all of our results. In this case, each f_i simply has domain $\{0,1\}^n$.

Product tests include as a special case several classes of tests which have been studied in the literature. Specifically, as mentioned in Definition 1, product tests include as a special case the important class of *combinatorial rectangles* [2, 42, 43, 37, 25, 8, 40, 52, 28, 31, 27].

▶ **Definition 2** (Combinatorial Rectangles)**.** A *combinatorial rectangle* is a product test where each f_i has output in $\{0,1\}$.

Product tests also include as a special case *combinatorial checkerboards* [53], corresponding to functions f_i with range $\{-1,1\}$. More generally, the recent work [27] highlights the unifying role of product tests (which are called Fourier shapes in [27]) by showing that any distribution that fools product tests also fools a number of other tests considered in the literature, including generalized halfspaces [30] and combinatorial shapes [29, 21]. For the main points in this paper it suffices to consider combinatorial rectangles, but we get broader results working with products.

Bounded independence vs. products. A moment's thought reveals that bounded independence completely fails to fool product tests. Indeed, note that the parity function on $m := nk$ bits is a product test: set $s = 2$ and let each of the k functions compute the parity of their n-bit input, with output in $\{-1,1\}$. However, consider the distribution D which is uniform on $m-1$ bits and has the last bit equal to the parity of the first $m-1$ bits. D has independence $m-1$, which is just one short of maximum. And yet the expectation of parity under D is 1, whereas the expectation of parity under uniform is 0.

This parity counterexample is the simplest example of a general obstacle which has more manifestations. For another example define $g_i := (1 - f_i)/2$, where the f_i are as in the previous example. Each g_i has range in $\{0,1\}$, and so $\prod g_i$ is a combinatorial rectangle. But the expectations of $\prod_i g_i$ under D and uniform differ by 2^{-k}. This error is too large for the applications in communication complexity and streaming where we have to sum over 2^k rectangles. Indeed, jumping ahead, having a much lower error is critical for our applications. Finally, the obstacle arises even if we consider distributions with small bias [41] instead of bounded independence. Indeed, the uniform distribution D over m bits whose inner product modulo 2 is one has bias $2^{-\Omega(m)}$, but inner product is a nearly balanced function which can be written as product, implying that the expectations under D and uniform differ by $1/2 - o(1)$.

The starting point of this work is the observation that all these examples break completely if we perturb just a few bits of D randomly. For parity, it suffices to perturb one bit and the expectation under D will be 0. For inner product, the distance between the expectations shrinks exponentially with the number of perturbed bits.

Our main result is that this is a general phenomenon: If we add a little noise to any distribution with bounded independence, or with small-bias, then we fool product tests with good error bounds. We first state the results for bounded independence. We begin with two definitions which are used extensively in this paper.

▶ **Definition 3.** A distribution D over $[s]^m$ is *b-wise independent*, or *b-uniform*, if any b symbols of D are uniformly distributed over $[s]^b$.

▶ **Definition 4** (Noise)**.** We denote by $E(s, m, \eta)$ the *noise distribution* over $[s]^m$ where the symbols are independent and each of them is set to uniform with probability η and is 0 otherwise. We simply write E when the parameters are clear from the context.

▶ **Theorem 5** (Bounded Independence Plus Noise Fools Products)**.** *Let $f_1, \ldots, f_k \colon [s]^n \to \mathbb{C}_1$ be k functions with $\mu_i = \mathrm{E}[f_i]$. Set $m := nk$ and let D be a b-uniform distribution over $[s]^m$. Let E be the noise distribution from Definition 4. Write $D = (D_1, D_2, \ldots, D_k)$ where each D_i is in $[s]^n$, and similarly for E. Then*

$$\left| \mathrm{E}\left[\prod_{i \le k} f_i(D_i + E_i) \right] - \prod_{i \le k} \mu_i \right| \le \varepsilon$$

for the following choices:
(1) *If $b \ge n$ then $\varepsilon = k(1-\eta)^{\Omega(b^2/m)}$.*
(2) *If $b < n$ and each D_i is uniform over $[s]^n$ then $\varepsilon = k(1-\eta)^{\Omega(b/k)}$.*
(3) *If $b < n$ then $\varepsilon = ke^{-\Omega(\eta b/k)} + 2k\binom{n}{n-b}e^{-\Omega(\eta b)}$.*
Moreover, there exist f_i and D such that $\varepsilon \ge (1-\eta)^b$. In particular, the bounds are tight up to the constants in the Ω when $b = \Omega(m)$ and $\eta \ge (\log k)/m$.

It is an interesting question whether the bounds are tight even for $b = o(m)$. We stress that the $D_i \in [s]^n$ in this theorem may not even be pairwise independent; only the m *symbols* of D are b-wise independent. We use (1) in most of our applications. Occasionally we use (3) with $b = n - 1$, in which case the error bound is $O(nke^{-\Omega(\eta n/k)})$.

Also note that the theorem is meaningful for a wide range of the noise parameter η: we can have η constant, which means that we are perturbing a constant fraction of the symbols, or we can have $\eta = O(1/m)$ which means that we are only perturbing a constant number of symbols, just like in the observation mentioned above. To illustrate this setting, consider for example $k = O(1)$ and $b = n$. We can have an error bound of ε by setting $\eta = c/m$ for a c that depends only on ε.

We now move to our results for small-bias distributions. A distribution D over m bits has bias δ if any parity of the bits (with range $\{-1, 1\}$) has expectation at most δ in magnitude. The following definition extends this to larger alphabets.

▶ **Definition 6.** A distribution $D = (D_1, D_2, \ldots, D_m)$ over $[s]^m$ is (b, δ)-biased if, for every nonzero $\alpha \in [s]^m$ with at most b non-zero coordinates we have $|\mathrm{E}_D[\omega^{\sum_i \alpha_i D_i}]| \le \delta$ where $\omega := e^{2\pi i/s}$. When $b = m$ we simply call D δ-biased.

In the case of small-bias distribution our bound on the error is a bit more complicated. We state next one possible tradeoff and defer a more general statement to §2.

▶ **Theorem 7.** *Let $f_1, \ldots, f_k \colon [s]^n \to \mathbb{C}_1$ be k functions with $\mu_i = \mathrm{E}[f_i]$. Assume $\delta \le s^{-n}$. Let D be an (n, δ)-biased distribution over $[s]^m$. Let E be the noise distribution from*

Definition 4. Write $D = (D_1, D_2, \ldots, D_k)$ *where each* D_i *is in* $[s]^n$*, and similarly for* E*. Then*

$$\left| \mathrm{E}\left[\prod_{i \le k} f_i(D_i + E_i) \right] - \prod_{i \le k} \mu_i \right| \le 2k(1-\eta)^{\Omega(\log(1/\delta)/(k \log sk))} + \sqrt{\delta}.$$

Note that [7] show that a (b, δ)-biased distribution over $\{0,1\}^m$ is ε-close in statistical distance to a b-uniform distribution, for $\varepsilon = \delta \sum_{i=1}^{b} \binom{m}{i}$. (See [4] for a similar bound.) One can apply their results in conjunction with Theorem 5 to obtain a result for small-bias distribution, but only if $\delta \le 1/\binom{m}{b}$. Via a direct proof we derive useful bounds already for $\delta = \Omega(2^{-b})$, and this will be used in §1.2.

We note that summing a noise vector to a string x is equivalent to taking a *random restriction* of x. With this interpretation our results show that on average a random restriction of a product test is a function f' that is simpler in the sense that f' is fooled by any (n, δ)-biased distribution, for certain values of δ. (The latter property has equivalent formulations in terms of the Fourier coefficients of f', see [11].) Thus, our results fall in the general theme "restrictions simplify functions" that has been mainstream in complexity theory since at least the work of Subbotovskaya [48]. For an early example falling in this theme, consider AC^0 circuits. There are distributions with super-constant independence which do not fool AC^0 circuits of bounded depth and polynomial size. (Take the uniform distribution conditioned on the parity of the first log many bits equal to 1, and use the fact that such circuits can compute parity on log many bits.) On the other hand, the *switching lemma* [26, 1, 54, 33, 34, 35] shows that randomly restricting all but a 1/polylog fraction of the variables collapses the circuit to a function that depends only on $c = O(1)$ variables, and such a function is fooled by any c-wise independent distribution. Thus, adding noise dramatically reduces the amount of independence that is required to fool AC^0 circuits. For a more recent example, Lemma 7.2 in [28] shows that for a special case of AC^0 circuits – read-once CNF – one can restrict all but a constant fraction of the variables and then the resulting function is fooled by any ε-bias distribution for a certain $\varepsilon = 1/n^{\omega(1)}$ which is seen to be larger than the bias that would be required had we not applied a restriction.

We are not aware of prior work which applies to arbitrary functions as in our theorems. Another difference between our results and all the previous works that we are aware of lies in the parameter η. In previous works η is large, in particular $\eta = \Omega(1)$, which corresponds to restricting many variables. We can instead set η arbitrarily, and this flexibility is used in both of our applications.

1.1 Application: The complexity of decoding

Error-correcting codes are a fundamental concept with myriad applications in computer science. It is relevant to several of these, and perhaps also natural, to ask what is the complexity of basic procedures related to error-correcting codes. In this paper we focus on *decoding*. The question of the complexity of decoding has already been addressed in [9, 10, 32]. However, all previous lower bounds that we are aware of are perhaps not as strong as one may hope. First, they provide no better negative results for decoding than for *encoding*. But common experience shows that decoding is much harder! Second, they do not apply to decision problems, but only to multi-output problems such as computing the entire message. Third, they apply to small-space algorithms but not to stronger models such as communication protocols.

In this work we obtain new lower bounds for decoding which overcome these limitations. First, we obtain much stronger bounds for decoding than for encoding. For example, we prove below that decoding a message symbol from Reed–Solomon codeword of length q with $\Omega(q)$ errors requires $\Omega(q)$ communication. On the other hand, encoding is a linear map, and so one can compute any symbol with just $O(\log q)$ communication (or space). This exponential gap may provide a theoretical justification for the common experience that decoding is harder than encoding. Second, our results apply to decision problems. Third, our results apply to stronger models than space-bounded algorithms. Specifically, our lower bounds are proved in the k-party "number-in-hand" communication complexity model, where each of k collaborating parties receives a disjoint portion of the input. The parties communicate by broadcast (a.k.a. writing on a blackboard). For completeness we give next a definition. Although we only define deterministic protocols, our lower bounds in fact bound the *correlation* between such protocols and the hard problem, and so also hold for distributions of protocols (a.k.a. allowing the parties to share a random string).

▶ **Definition 8** (Number-in-Hand Protocols)**.** A *k-party number-in-hand, best-partition, communication protocol* for a function $f\colon [s]^m \to Y$, where k divides m, is given by a partition of m into k sets $S_1, S_2, \ldots, S_k$ of equal size m/k and a binary tree. Each internal node v of the tree is labeled with a set $S_v \in \{S_1, S_2, \ldots, S_k\}$ and a function $f_v\colon [s]^{m/k} \to \{0,1\}$, and has two outgoing edges labeled 0 and 1. The leaves are labeled with elements from Y. On input $x \in [s]^m$ the protocol computes $y \in Y$ following the root-to-leaf path where from node v we follow the edge labeled with the value of f_v on the m/k symbols of x corresponding to S_v. The communication cost of the protocol is the depth of the tree.

Note that we insisted that k divides m, but all the results can be generalized to the case when this does not hold. However this small additional generality makes the statements slightly more cumbersome, so we prefer to avoid it. Jumping ahead, for Reed–Solomon codes this will mean that the claims do not apply as stated to prime fields (but again can be modified to apply to such fields).

Again for completeness, we give next a definition of space-bounded algorithms. For simplicity we think of the input as being encoded in bits.

▶ **Definition 9** (One-Way, Bounded-Space Algorithm)**.** A *width-W (a.k.a. space-$\log W$) one-way algorithm* (or *branching program* or *streaming algorithm*) on m bits consists of a layered directed graph with $m+1$ layers. Each layer has W nodes, except the first layer, which has 1 node, and the last layer, which has $2W$. Each node in layer $i \le m$ has two edges, labeled with 0 and 1, connecting to nodes in layer $i+1$. Each node on layer $m+1$ is labeled with an output element. On an m-bit input, the algorithm follows the path corresponding to the input, reading the input in a one-way fashion (so layer i reads the i-th input bit), and then outputs the label of the last node.

We note that a space-s one-way algorithm can be simulated by a k-party protocol with communication sk. Thus our negative results apply to space-bounded algorithms as well. In fact, this simulation only uses one-way communication and a fixed partition (corresponding to the order in which the algorithm reads the input). But our communication lower bounds hold even for two-way communication and for any partition of the input into k parties, as in Definition 8.

Our lower bound holds when the uniform distribution over the code is b-uniform.

▶ **Definition 10.** A code $C \subseteq \mathbb{F}_q^m$ is b-uniform if the uniform distribution over C is b-uniform.

The following standard fact relates the above definition to the *dual distance* of the code.

▶ **Fact 11.** *Let X be the uniform distribution over a linear code $C \subseteq \mathbb{F}_q^m$. Then X is d-wise independent if and only if the dual of C has minimum distance $\geq d+1$.*

We state next a lower bound for distinguishing a noisy codeword from uniform. The "-1" in the assumption on b will be useful later.

▶ **Theorem 12** (Distinguishing Noisy Codewords from Uniform is Hard)**.** *Let $C \subseteq \mathbb{F}_q^m$ be a b-uniform code. Let E be the noise distribution from Definition 4. Let k be an integer dividing m such that $b \geq m/k - 1$. Let $P\colon \mathbb{F}_q^m \to \{0,1\}$ be a k-party protocol using c bits of communication. Then*

$$|\Pr[P(C+E)=1] - \Pr[P(U)=1]| \leq \varepsilon \quad \text{for} \quad \varepsilon = 2^{c+\log(m)+O(1)-\Omega(\eta b^2/m)},$$

where C and U denote the uniform distributions over the code C and $\mathbb{F}_q^m$ respectively.

We now make some remarks on this theorem. First, we note that a (ck)-party protocol can be simulated by a k-party protocol, so in this sense the lower the number of parties the stronger the lower bound. Also, the smallest number of parties to which the theorem can apply is $k = m/b$, because for $k = m/b - 1$ one can design b-uniform codes such that the distribution $C + E$ can be distinguished well from uniform by just one party, cf. §A. And our lower bound applies for that number. The theorem is non-trivial whenever $b = \omega(\sqrt{m})$, but we illustrate it in the setting of $b = \Omega(m)$ which is typical in coding theory as we are also going to discuss. In this setting we can also set $k = m/b = O(1)$. Hence the communication lower bound is

$$c \geq \Omega(\eta m)$$

when $\eta \geq C \log m/m$ for a universal constant C. When $\eta = \Omega(1)$ this becomes $\Omega(m)$. Note that this bound is within an $O(\log q)$ factor of the bit-length of the input, which is $O(m \log q)$, and within a constant factor if $q = O(1)$.

We prove an essentially matching upper bound in terms of η, stated next. The corresponding distinguisher is a simple variant of *syndrome decoding* which we call "truncated syndrome decoding." It can be implemented as a small-space algorithm with one-sided error, and works even against adversarial noise. So the theorems can be interpreted as saying that syndrome decoding uses an optimal amount of space. We denote by $V(t)$ the volume of the q-ary Hamming ball in m dimensions of radius t, i.e., the number of $x \in \mathbb{F}_q^m$ with at most t non-zero coordinates.

▶ **Theorem 13** (Truncated Syndrome Decoding)**.** *Let $C \subseteq \mathbb{F}_q^m$ be a linear code with dimension d. Given t and $\delta > 0$ define $s := \lceil \log_q(V(t)/\delta) \rceil$. If $d \leq m - s$ there is a one-way algorithm A that runs in space $s \log q$ such that*
(1) *for every $x \in C$ and for every e of Hamming weight $\leq t$, $A(x+e) = 1$, and*
(2) *$\Pr[A(U)=1] \leq \delta$, where U is uniform in $\mathbb{F}_q^m$.*
Moreover, the space bound $s \log q$ is at most $O(t \log(mq/t)) + \log 1/\delta$.

Note that when $t = O(\eta m)$ and δ is constant the space bound is $O(\eta m \log(q/\eta))$, which matches our $\Omega(\eta m)$ lower bound up to the $O(\log(q/\eta))$ factor.

These results in particular apply to Reed–Solomon codes. Recall that a Reed–Solomon code of dimension b is the linear code where a message in $\mathbb{F}_q^b$ is interpreted as a polynomial p of degree $b-1$ and encoded as the q evaluations of p over any element in the field. (In some presentations, the element 0 is excluded.) Such a code is b-uniform because for any b points

(x_i, y_i) where the x_i's are different, there is exactly one polynomial p of degree $b-1$ such that $p(x_i) = y_i$ for every i.

For several binary codes $C \subseteq \mathbb{F}_2^m$ and constant η we can obtain a communication lower bound of $\Omega(m)$ which is tight up to constant factors. This is true for example for random, linear codes (with bounded rate). The complexity of decoding such codes is intensely studied, also because the assumed intractability of their decoding is a basis for several cryptographic applications. See for example [12], a slight improvement on the running time which already has more than 100 citations. We also obtain a tight lower bound of $\Omega(m)$ for several explicitly-defined binary codes. For example, we can pick an explicit binary code $C \subseteq \mathbb{F}_2^m$ which is $\Omega(m)$-uniform and that can be decoded in polynomial time for a certain constant noise parameter η (with high probability), see [46] for a construction.

Lower bounds for decoding one symbol. The lower bound in Theorem 12 is for the problem of distinguishing noisy codewords from uniform. Intuitively, this is a strong lower bound saying that no bit of information can be obtained from a noisy codeword. We next use this result to obtain lower bounds for decoding one symbol of the message given a noisy codeword. Some care is needed because some message symbols may just be copied in the codeword. This would allow one party to decode those symbols with no communication, even though the noisy codeword may be indistinguishable from uniform. The lower bound applies to codes that remain b-uniform even after fixing some input symbol. For such codes, a low-communication protocol cannot decode that symbol significantly better than by guessing at random.

▶ **Theorem 14.** *Let $C' \subseteq \mathbb{F}_q^m$ be a linear code with an $m \times r$ generator matrix G. Let $i \in \{1, 2, \ldots, r\}$ be an index, and let C be the code defined as $C := \{Gx \mid x_i = 0\}$. Let E be the noise distribution from Definition 4. Let k be an integer. Suppose that C is b-uniform for $b \geq m/k - 1$. Let $P \colon \mathbb{F}_q^m \to \mathbb{F}_q$ be a k-party protocol using c bits of communication. Then*

$$\Pr[P(GU + E) = U_i] \leq 1/q + \varepsilon,$$

where $U = (U_1, U_2, \ldots, U_r)$ is the uniform distribution and ε is as in Theorem 12.

We remark that whether C is b-uniform in general depends on both G and i. For example, let C' be a Reed–Solomon code of dimension $b = m/k$. Recall that C' is b-uniform. Note that if we choose $i = 0$ (corresponding to the evaluation of the polynomial at the point $0 \in \mathbb{F}_q$, which as we remarked earlier is a point we consider) then C has a fixed symbol and so is not even 1-uniform. On the other hand, if $i = b - 1$ then we obtain a Reed–Solomon code with dimension $b - 1$, which is $(b-1)$-uniform, and the lower bound in Theorem 14 applies.

We again obtain an almost matching upper bound. In fact, the corresponding protocol recovers the entire message.

▶ **Theorem 15** (Recovering Messages from Noisy Codewords). *Let $C \subseteq \mathbb{F}_q^m$ be a code with distance d. Let t be an integer such that $2t < d$, and let k be an integer dividing q.*

There is a k-party protocol $P \colon \mathbb{F}_q^m \to \mathbb{F}_q^b$ communicating $\max\{m - d + 2t + 1 - m/k, 0\} \lceil \log_2 q \rceil$ bits such that for every $x \in C$ and every e of Hamming weight $\leq t$, $P(x + e) = x$.

A Reed–Solomon code with dimension b has distance $d = m - b + 1$. Hence we obtain communication $\max\{b + 2t - m/k, 0\} \lceil \log_2 q \rceil$, for any t such that $2t < m - b + 1$. This upper bound matches the lower bound in Theorem 14 up to a $\log q$ factor. For example, when $k = O(1)$ and $b = q/k$ our upper bound is $O(\eta q \log q)$ and our lower bound is $\Omega(\eta q) - O(\log q)$.

1.2 Application: Pseudorandomness

The construction of explicit *pseudorandom generators* against restricted classes of tests is a fundamental challenge that has received a lot of attention at least since the 80's, cf. [3, 2]. One class of tests extensively considered in the literature is concerned with algorithms that read the input bits in a *one-way fashion in a fixed order*. A leading goal is to prove RL=L by constructing generators with logarithmic seed length that fool one-way, space-bounded algorithms, but here the seminal papers [42, 37, 43] remain the state of the art and have larger seed lengths. However, somewhat better generators have been obtained for several special cases, including for example combinatorial rectangles [2, 42, 43, 37, 25, 8, 40, 52, 28, 31], combinatorial shapes [29, 21, 27], and product tests [27]. In particular, for combinatorial rectangles $f\colon (\{0,1\}^n)^k \to \{0,1\}$ two incomparable results are known. For context, the minimal seed length up to constant factors is $O(n+\log(k/\varepsilon))$. One line of research culminating in [40] gives generators with seed length $O(n+\log k+\log^{3/2}(1/\varepsilon))$. More recently, [28] (cf. [31]) improve the dependence on ε while making the dependence on the other parameters a bit worse: they achieve seed length $O((\log n)(n+\log(k/\varepsilon)))+O(\log(1/\varepsilon)\log\log(1/\varepsilon)\log\log\log(1/\varepsilon))$. The latter result is extended to products in [27] (with some other lower-order losses).

Recently there has been considerable interest in extending tests by allowing them to read the bits in *any order*: [13, 14, 36, 45, 47]. This extension is significantly more challenging, and certain instantiations of generators against one-way tests are known to fail [13].

We contribute new pseudorandom generators that fool product tests in any order.

▶ **Definition 16** (Fooling)**.** A generator $G\colon \{0,1\}^\ell \to \{0,1\}^m$ *ε-fools* (or *fools with error ε*) a class T of tests on m bits if for every function $f \in T$ we have $|\mathrm{E}[f(G(U_\ell)) - \mathrm{E}[f(U_m)]| \le \varepsilon$, where U_ℓ and U_m are the uniform distributions on ℓ and m bits respectively. We call ℓ the seed length of G. We call G explicit if it is computable in time polynomial in m.

▶ **Definition 17** (Any order)**.** We say that a generator $G\colon \{0,1\}^\ell \to \{0,1\}^m$ ε-fools a class T of tests *in any order* if for every permutation π on m bits the generator $\pi \circ G\colon \{0,1\}^\ell \to \{0,1\}^m$ ε-fools T.

The next theorem gives some of our generators. The notation $\tilde{O}()$ hides logarithmic factors in k and n. In this section we only consider alphabet size $s = 2$. We write the range $\{0,1\}^{nk}$ of the generators as $(\{0,1\}^n)^k$ to indicate the parameters of the product tests.

▶ **Theorem 18** (PRG for Any-Order Products, I)**.** *There exist explicit pseudorandom generators $G\colon \{0,1\}^\ell \to (\{0,1\}^n)^k$ that ε-fool product tests in any order, with the following seed lengths:*
(1) *$\ell = 2n + O(k^2 \log k \log(k/\varepsilon)\log n) = 2n + \tilde{O}(k^2\log(1/\varepsilon))$, and*
(2) *$\ell = O(n) + O(n^{2/3}(k^2\log k\log(k/\varepsilon)\log n)^{1/3}) = O(n) + \tilde{O}((nk)^{2/3}\log^{1/3}(1/\varepsilon))$.*
Moreover, the generators' output has the form $D + E'$, where D is a small-bias distribution and E' is statistically close to a noise vector.

One advantage of these generators is their simplicity. Constructions in the literature tend to be somewhat more involved. In terms of parameters, we note that when $k = O(1)$ we achieve in (1) seed length $\ell = 2n+O(\log 1/\varepsilon)\log n$, which is close to the value of $n+O(\log 1/\varepsilon)$, which is optimal even for the case of fixed order and $k = 2$. Our result is significant already for $k = 3$, but not for $k = 2$. In the latter case the seed length of $(2-\Omega(1))n$ obtained in [13] remains the best known. For $k \ge \sqrt{n}$ our generator in (2) has polynomial stretch, using a seed length $\tilde{O}(m^{2/3})$ for output length m.

We note that for the special case of combinatorial rectangles $f\colon (\{0,1\}^n)^k \to \{0,1\}$ a pseudorandom generator with seed length $O((n+\log k)\log(1/\varepsilon))$ follows from previous work.

The generator simply outputs m bits such that any $d \cdot n$ of them are $1/k^d$ close to uniform in statistical distance, where $d = c\log(1/\varepsilon)$ for an appropriate constant c. Theorem 3 in [6] shows how to generate these bits from a seed of length $O(n\log(1/\varepsilon) + \log\log m + \log(1/\varepsilon)\log k) = O((n+\log k)\log(1/\varepsilon))$. The analysis of this generator is as follows. The induced distribution on the outputs of the f_i is a distribution on $\{0,1\}^k$ such that any d bits are $1/k^d$ close to the distribution of independent variables whose expectations are equal to the $\mathrm{E}[f_i]$. Now Lemma 5.2 in [18] (cf. [24]) shows that the probability that the And of the output is 1 equals the product of the expectations of the f_i plus an error which is $\leq 2^{-\Omega(d)} + d\binom{k}{d}/k^d \leq \varepsilon$. However this generator breaks down if the output of the functions is $\{-1,1\}$ instead of $\{0,1\}$. Moreover, its parameters are incomparable with those in Theorem 18.(I). In particular, for $k = O(1)$ its seed length is $\geq n\log(1/\varepsilon)$, while as remarked above we achieve $O(n + \log(n)\log(1/\varepsilon))$.

We are able to improve the seed length of (2) in Theorem 18 to $\tilde{O}(\sqrt{m})$, but then the resulting generator is more complicated and in particular it does not output a distribution of the form $D + E'$. For this improvement we "derandomize" our theorems 5 and 7 and then combine them with a recursive technique originating in [28] (cf. [3]) and used in several subsequent works including [45, 47, 19]. Our context and language are somewhat different from previous work, and this fact may make this paper useful to readers who wish to learn the technique.

▶ **Theorem 19** (PRG for Any-Order Products, II). *There exists an explicit pseudorandom generator $G\colon \{0,1\}^\ell \to (\{0,1\}^n)^k$ that ε-fools product tests in any order and seed length $\ell = O(n + \sqrt{nk\log k\log(k/\varepsilon)}) = O(n) + \tilde{O}(\sqrt{nk\log 1/\varepsilon})$.*

Recall that for $b = n$ the error bound in our Theorem 5 is $k(1-\eta)^{\Omega(b/k)}$, and that it is open whether the exponent can be improved to $\Omega(b)$. We show that if such an improvement is achieved for the derandomized version of the theorem (stated later in Theorem 37) then one would get much better seed length: $\ell = O((n + \log k\log(m/\varepsilon))\log m)$.

Reingold, Steinke, and Vadhan [45] give a generator that ε-fools width-W space algorithms on m bits in any order, with seed length $\ell = \tilde{O}(\sqrt{m}\log(W/\varepsilon))$. Every combinatorial rectangle $f\colon (\{0,1\}^n)^k \to \{0,1\}$ can be computed by a one-way algorithm with width $2^{n-1}+1$ on $m = nk$ bits. Hence they also get seed length $\tilde{O}(\sqrt{nk}(n + \log 1/\varepsilon))$ for combinatorial rectangles. Our Theorem 19 improves upon this by removing a factor of n.

Going in the other direction, if D is a distribution on $(\{0,1\}^n)^k$ bits that ε-fools combinatorial rectangles, then D also fools width-W one-way algorithms on $m = nk$ bits with error $W^k\varepsilon$.

▶ **Corollary 20** (Bounded Independence Plus Noise Fools Space). *Let D be a b-uniform distribution on m bits. Let E be the noise distribution from Definition 4. If $b \geq m^{2/3}\log m$ and η is any constant then $D + E$ fools $O(\log m)$-space algorithms in any order with error $o(1)$.*

As mentioned earlier, [27] show that if a generator fools products then it also fools several other computational models, with some loss in parameters. As a result, we obtain generators for the following two models, extended to read bits in any order.

▶ **Definition 21** (Generalized halfspaces and combinatorial shapes). A *generalized halfspace* is a function $h\colon (\{0,1\}^n)^k \to \{0,1\}$ defined by $h(x) := 1$ if and only if $\sum_{i\leq k} g_i(x_i) \geq \theta$, where $g_1,\ldots,g_k\colon \{0,1\}^n \to \mathbb{R}$ are arbitrary functions and $\theta \in \mathbb{R}$.

A *combinatorial shape* is a function $f\colon (\{0,1\}^n)^k \to \{0,1\}$ defined by $f(x) := g(\sum_{i\leq k} g_i(x_i))$, where $g_1,\ldots,g_k\colon \{0,1\}^n \to \{0,1\}$ and $g\colon \{0,\ldots,k\} \to \{0,1\}$ are arbitrary functions.

▶ **Theorem 22** (PRG for Generalized Halfspaces and Combinatorial Shapes, in Any-Order)**.** *There exists an explicit pseudorandom generator* $G\colon \{0,1\}^\ell \to (\{0,1\}^n)^k$ *that* ε*-fools both generalized halfspaces and combinatorial shapes in any order with seed length* $\ell = \tilde{O}(n\sqrt{k} + \sqrt{nk\log(1/\varepsilon)})$.

Note that for $\varepsilon = 2^{-O(n)}$ the seed length simplifies to $\tilde{O}(n\sqrt{k})$.

An original motivation for this work is the study of the sum of small-bias distributions [38]. However the relationship between the results in this work and [38] is somewhat technical, applying only to certain settings of parameters. Hence we defer the discussion to §6.

1.3 Techniques

We now give an overview of the proof of Theorem 5. The natural high-level idea, which our proof adopts as well, is to apply Fourier analysis and use noise to bound high-degree terms and independence to bound low-degree terms. Part of the difficulty is finding the right way to decompose the product $\prod_{i\le k} f_i$. We proceed as follows. For a function f let f^H be its "high-degree" Fourier part and f_L be its "low-degree" Fourier part, so that $f = f^H + f^L$. Our goal is to go from $\prod f_i$ to $\prod f_i^L$. The latter is a product of low-degree functions and hence has low degree. Therefore, its expectation will be close to $\prod_i \mu_i$ by the properties of the distribution D; here we do not use the noise E.

To move from $\prod f_i$ to $\prod f_i^L$ we pick one f_j and we decompose it as $f_j^H + f_j^L$. Iterating this process we indeed arrive to $\prod f_i^L$, but we also obtain k extra terms of the form

$$f_1 f_2 \dots f_{j-1} f_j^H f_{j+1}^L f_{j+2}^L \dots f_k^L$$

for $j = 1, \dots, k$. We show that each of these terms is close to 0 thanks to the presence of the high-degree factor f_j^H. Here we use both D and E.

We conclude this section with a brief technical comparison with the recent papers [28, 31, 27] which give generators for combinatorial rectangles (and product tests). We note that the generators in those papers only fool tests $f = f_1 \cdot f_2 \cdots f_k$ that read the input in a fixed order (whereas our results allow for any order). Also, they do not use noise, but rather hash the functions f_i in a different way. Finally, a common technique in those papers is, roughly speaking, to use hashing to *reduce the variance of the functions*, and then show that bounded independence fools functions with small variance. We note that the noise parameters we consider in this work are too small to be used to reduce the variance. Specifically, for a product test f those papers define a new function $g = g_1 \cdot g_2 \cdots g_k$ which is the average of f over t independent inputs. While g has the same expectation as f, the variance of each g_i is less than that of f_i by a factor of t. Their goal is to make the variance of each g_i less than $1/k$ so that the sum of the variances is less than 1. In order to achieve this reduction with noise we would have to set $\eta \ge 1 - 1/\sqrt{k}$. This is because if f_i simply is $(-1)^x$ where x is one bit, then the variance of f_i perturbed by noise is $\mathrm{E}_x[\mathrm{E}_E^2[(-1)^{x+E}]] - \mathrm{E}_{x,E}[(-1)^{x+E}] = \mathrm{E}_{x,E,E'}[(-1)^{E+E'}] = (1-\eta)^2$.

Organization. In §2 we prove our main theorems, 5 and 7. In §3 we give the proof details for the results in §1.1. The details for the results in §1.2 are spread over three sections. In §4 we prove Theorem 18. In §5 we prove Theorem 19, and discuss the potential improvement. In §6 we prove Theorem 22, and discuss the relationship between this paper and an original motivation [38]. We conclude in §7. In §A we include for completeness a lower bound on the values of b and η for which Theorem 5 can apply.

2 Bounded independence plus noise fools products

In this section we prove Theorem 5 and Theorem 7. They both follow easily from the next theorem which is the main result in this section.

▶ **Theorem 23.** *Let $t \in [0,n]$. Let $f_1,\ldots,f_k\colon [s]^n \to \mathbb{C}_1$ be k functions with $\mu_i = \mathrm{E}[f_i]$. Let D be a (b,δ)-biased distribution over $[s]^m$ for $b \geq \max\{n, 2(k-1)t\}$. Let E be the noise distribution from Definition 4. Write $D = (D_1, D_2, \ldots, D_k)$ where each D_i is in $[s]^n$, and similarly for E. Then*

$$\left| \mathrm{E}\left[\prod_{i\leq k} f_i(D_i + E_i)\right] - \prod_{i\leq k}\mu_i \right| \leq k(1-\eta)^t \sqrt{(1+s^n\delta)(1+V(t)^{k-1}\delta)} + V(t)^{k/2}\delta.$$

Let us quickly derive Theorem 5 and 7 in the introduction.

Proof of Theorem 5. Setting $\delta = 0$ and $t = b/2(k-1)$ in Theorem 23 gives the bound

$$k(1-\eta)^{b/2(k-1)} \qquad (\star)$$

which proves the theorem in the case $n \leq b = O(n)$.

To prove (1) we need to handle larger b. For this, let $c := \lfloor b/n \rfloor$, and group the k functions into $k' \leq k/c + 1$ functions on input length $n' := cn$. Note that $b \geq n'$, and so we can apply $(\star)$ to

$$k'(1-\eta)^{\Omega(b/k')} \leq k(1-\eta)^{\Omega(b^2/kn)}.$$

To prove (2) one can observe that in the proof of $(\star)$ the condition $b \geq n$ is only used to guarantee that each D_i is uniform. The latter is now part of our assumption.

To prove (3) view the noise vector E as the sum of two noise vectors E' and E'' with parameter α such that $1-\eta = (1-\alpha)^2$. Note this implies $\alpha = \Omega(\eta)$. If E' sets to uniform at least $n-b$ coordinates in each function then we can apply $(\star)$ to functions on $\leq b$ symbols with η replaced by α. The probability that E' does not set to uniform that many coordinates is at most

$$k\binom{n}{n-b}(1-\alpha)^b \leq k\binom{n}{n-b}e^{-\Omega(\eta b)},$$

and in that case the distance between the expectations is at most two.

To show the "moreover" part let the f_i compute parity on the first $b+1$ bits, and let D be the b-wise independent distribution which is uniform on strings whose parity of the $b+1$ bits is 0. The other bits are irrelevant. The expectation of parity under uniform is 0. The expectation of parity under D is 1 if no symbol is perturbed with noise, and is 0 otherwise. Hence the error is $\geq (1-\eta)^{b+1}$. In particular, if $b = \Omega(m)$ then an upper bound on the error of the form $k(1-\eta)^{cm}$ is false for sufficiently large c, using that $\eta \geq (\log k)/m$. ◀

Proof of Theorem 7. Let $c := \lfloor \sqrt{\log(1/\delta)/(n\log s)} \rfloor$. Note that $c \geq 1$ because $\delta \leq s^{-n}$. We group the k functions into $k' = \lceil k/c \rceil$ functions on input length $n' := cn$. The goal is to make $s^{n'} \approx 1/\delta$. By Claim 33, $V_{n'}(t) \leq (en's/t)^t$. Hence $V_{n'}(t)^{k'/2} \leq V_{n'}(t)^{k'-1} \leq (en's/t)^{k't}$. Now let $t = \alpha n' \log s/(k' \log sk')$ for a small constant $\alpha > 0$ so that the latter bound is $\leq s^{n'/2} \approx 1/\sqrt{\delta}$.

The error bound in Theorem 23 now becomes at most

$$k(1-\eta)^t(1+s^{n'}\delta) + s^{n'/2}\delta.$$

And so the bound is at most

$$2k(1-\eta)^{\Omega(\log(1/\delta)/(k\log sk))} + \sqrt{\delta}. \qquad \blacktriangleleft$$

We now turn to the proof of Theorem 23. We begin with some preliminaries.

2.1 Preliminaries

Denote by U the uniform distribution. Let s be any positive integer. We write $[s]$ for $\{0, 1, 2, \ldots, s-1\}$. Let $\omega := e^{2\pi i/s}$ be a primitive s-th root of unity. For any $\alpha \in [s]^u$, we define $\chi_\alpha(x)\colon [s]^u \to \mathbb{C}$ to be

$$\chi_\alpha(x) := \omega^{\langle \alpha, x\rangle},$$

where α and x are viewed as vectors in $\mathbb{Z}_s^u$ and $\langle \alpha, x\rangle := \sum_i \alpha_i x_i$.

For any function $f\colon [s]^u \to \mathbb{C}$, its Fourier expansion is

$$f(x) := \sum_{\alpha \in [s]^u} \hat{f}_\alpha \chi_\alpha(x),$$

where $\hat{f}_\alpha \in \mathbb{C}$ is given by

$$\hat{f}_\alpha := \mathop{\mathrm{E}}_{x\sim[s]^u}[f(x)\overline{\chi_\alpha(x)}].$$

Here and elsewhere, random variables are uniformly distributed unless specified otherwise.

The Fourier L_1-*norm* of f is defined as $\sum_\alpha |f_\alpha|$, and is denoted by $L_1[f]$. The *degree* of f is defined as $\max\{|\alpha| : \hat{f}_\alpha \neq 0\}$, where $|\alpha|$ is the number of nonzero coordinates of α, and is denoted by $\deg(f)$. Note that we have $L_1[f] = L_1[\overline{f}]$. The following fact bounds the L_1-norm and degree of product functions.

▶ **Fact 24.** *For any two functions $f, g\colon [s]^u \to \mathbb{C}$, we have*
(1) $\deg(fg) \le \deg(f) + \deg(g)$*, and*
(2) $L_1[fg] \le L_1[f]L_1[g]$.

Proof. We have

$$f(x)g(x) = \left(\sum_{\alpha\in[s]^n} \hat{f}_\alpha \chi_\alpha(x)\right)\left(\sum_{\beta\in[s]^n} \hat{g}_\beta \chi_\beta(x)\right)$$
$$= \sum_{\alpha,\beta} \hat{f}_\alpha \hat{g}_\beta \chi_{\alpha+\beta}(x) = \sum_\alpha \left(\sum_\beta \hat{f}_{\alpha-\beta}\hat{g}_\beta\right)\chi_\alpha(x).$$

Hence the α-th Fourier coefficient of $f\cdot g$ is $\sum_\beta \hat{f}_{\alpha-\beta}\hat{g}_\beta$.

To see (1), note that in the latter expression the sum over β can be restricted to those β with $|\beta| \le \deg(g)$. Now note that if $|\alpha| > \deg(f) + \deg(g)$ then $|\alpha - \beta| > \deg(f)$ and hence $\hat{f}_{\alpha-\beta}$ will be zero for every β.

To show (2) write $L_1[fg] = \sum_\alpha |\sum_\beta \hat{f}_{\alpha-\beta}\hat{g}_\beta| \le \sum_{\alpha,\beta} |\hat{f}_{\alpha-\beta}||\hat{g}_\beta| = (\sum_\alpha |\hat{f}_\alpha|)(\sum_\beta |\hat{g}_\beta|) = L_1[f]L_1[g]$. ◀

▶ **Fact 25** (Parseval's Identity)**.** $\sum_{\alpha\in[s]^n} |\hat{f}_\alpha|^2 = \mathrm{E}_{x\sim[s]^n}[|f(x)|^2]$. *In the case of $f \in \mathbb{C}_1$, this quantity is at most* 1.

Proof.

$$\mathop{\mathrm{E}}_{x\sim[s]^n}[f(x)\overline{f(x)}] = \mathop{\mathrm{E}}_{x\sim[s]^n}[\sum_{\alpha\in[s]^n}\hat{f}_\alpha\chi_\alpha(x)\cdot\overline{\sum_{\alpha'\in[s]^n}\hat{f}_{\alpha'}\chi_{\alpha'}(x)}]$$
$$= \sum_{\alpha,\alpha'\in[s]^n}\hat{f}_\alpha\overline{\hat{f}_{\alpha'}}\mathop{\mathrm{E}}_{x\sim[s]^n}[\chi_{\alpha-\alpha'}(x)] = \sum_{\alpha\in[s]^n}|\hat{f}_\alpha|^2.$$

where the last equality holds because we have $\mathrm{E}_{x\sim[s]^n}[\chi_{\alpha-\alpha'}(x)]$ equals 0 if $\alpha\neq\alpha'$ and equals 1 otherwise. ◀

▶ **Fact 26.** *Let $E=(E_1,\ldots,E_k)$ be the distribution over $[s]^k$, where the symbols are independent and each of them is set to uniform with probability η and is 0 otherwise. Then for every $\alpha\in[s]^n$, $\mathrm{E}[\chi_\alpha(E)]=(1-\eta)^{|\alpha|}$.*

Proof. The expectation conditioned on the event "E sets none of the nonzero positions of α to uniform" is 1. This event happens with probability $(1-\eta)^{|\alpha|}$. Conditioned on its complement, the expectation is 0. To see this, assume that the noise vector sets to uniform position i of α, and that $\alpha_i\neq 0$. Let $\beta:=\omega^{\alpha_i}$. Then the expectation can be written as a product where a factor is

$$\mathop{\mathrm{E}}_{x\sim\{0,1,\ldots,s-1\}}[\beta^x]=\frac{1}{s}\cdot\frac{\beta^s-1}{\beta-1}=0,$$

using the fact that $\beta\neq 1$ because $\alpha_i\in\{1,2,\ldots,s-1\}$ and that $\beta^s=(\omega^s)^{\alpha_i}=1$. Therefore the total expectation is $(1-\eta)^{|\alpha|}$. ◀

Note that this lemma includes the uniform $\eta=1$ case, with the convention $0^0=1$.

We will use the following facts multiple times.

▶ **Fact 27.** *Let $f\colon[s]^n\to\mathbb{C}$ be a function with degree b. We have:*
(1) *For any (b,δ)-biased distribution D over $[s]^n$, $|\mathrm{E}[f(D)]-\mathrm{E}[f(U)]|\le L_1[f]\delta$,*
(2) *For any $(2b,\delta)$-biased distribution D over $[s]^n$, $|\mathrm{E}[|f(D)|^2]-\mathrm{E}[|f(U)|^2]|\le L_1[f]^2\delta$, and*
(3) *the bound in (2) holds even if D is (n,δ) biased.*

Proof. For (1), note that

$$|\mathrm{E}[f(D)]-\mathrm{E}[f(U)]| = \left|\sum_{0<|\alpha|\le b}\hat{f}_\alpha\,\mathrm{E}[\chi_\alpha(D)]\right| \le \sum_{0<|\alpha|\le b}|\hat{f}_\alpha||\mathrm{E}[\chi_\alpha(D)]| \le L_1[f]\delta.$$

For (2), recall that $|f(x)|^2=f(x)\overline{f(x)}$. By Fact 24, the function $|f(x)|^2$ has degree $\le 2b$. Also, again by Fact 24 the L_1-norm of that function is at most $L_1[f]\cdot L_1[\bar{f}]=L_1[f]^2$. Now the result follows by (1).

Finally, (3) is proved like (2), noting that a function on $[s]^n$ always has degree $\le n$. ◀

Actually the bounds hold with $\sum_{\alpha\neq 0}|\hat{f}_\alpha|$ instead of $L_1[f]$, but we will not use that.

2.2 Proof of Theorem 23

For a function $f\colon[s]^n\to\mathbb{C}_1$, consider its Fourier expansion $f(x):=\sum_\alpha\hat{f}_\alpha\chi_\alpha(x)$, and let $f^L(x):=\sum_{\alpha:|\alpha|\le t}\hat{f}_\alpha\chi_\alpha(x)$ and $f^H(x):=\sum_{\alpha:|\alpha|>t}\hat{f}_\alpha\chi_\alpha(x)$. Define $F_i\colon([s]^n)^k\to\mathbb{C}$ to be

$$F_i(x_1,\ldots,x_k):=\left(\prod_{j<i}f_j(x_j)\right)\cdot f_i^H(x_i)\cdot\left(\prod_{\ell>i}f_\ell^L(x_\ell)\right).$$

Pick f_k and write it as $f_k^L + f_k^H$. We can then rewrite

$$\prod_{1\le i\le k} f_i = F_k + \left(\prod_{1\le i\le k-1} f_i\right)\cdot f_k^L.$$

We can reapply the process to $(\prod_{1\le i\le k-1} f_i)$. Continuing this way, we eventually have what we want to bound, i.e. $|\mathrm{E}[\prod_{i\le k} f_i(D_i+E_i)] - \prod_{i\le k}\mu_i|$, is at most

$$\left|\sum_{i\le k}\mathrm{E}[F_i(D+E)]\right| + \left|\mathrm{E}[\prod_{i\le k} f_i^L(D_i+E_i)] - \prod_{i\le k}\mu_i\right|.$$

The theorem follows readily from the next two lemmas, the second of which has a longer proof.

▶ **Lemma 28.** $|\mathrm{E}[\prod_{i\le k} f_i^L(D_i+E_i)] - \prod_{i\le k}\mu_i| \le V(t)^{k/2}\delta$.

Proof. Fix E arbitrarily. Each f_i^L has degree at most t, and by the Cauchy–Schwarz inequality, it has L_1-norm $\sum_{|\alpha|\le t}|\hat{f}_\alpha| \le V(t)^{1/2}(\sum_\alpha|\hat{f}_\alpha|^2)^{1/2} \le V(t)^{1/2}$. Here we use the fact that f maps to $\mathbb{C}_1$ and Fact 25. Hence, by Fact 24, $\prod_{0<i\le k} f_i^L$ has degree at most kt and L_1-norm at most $V(t)^{k/2}$. By hypothesis, D is (b,δ)-biased, and this also holds for $D+E$ for any fixed E. Moreover, $b \ge 2(k-1)t \ge kt$, and so by (1) in Fact 27 we have

$$\left|\mathop{\mathrm{E}}_D[\prod_{i\le k} f_i^L(D_i+E_i)] - \prod_{i\le k}\mu_i\right| \le V(t)^{k/2}\delta.$$

Averaging over E proves the claim. ◀

▶ **Lemma 29.** *For every $i \in \{1,2,\dots,k\}$, we have*

$$|\mathrm{E}[F_i(D+E)]| \le (1-\eta)^t\sqrt{(1+s^n\delta)(1+V(t)^{k-1}\delta)}.$$

Proof. We have

$$\begin{aligned}
|\mathrm{E}[F_i(D+E)]| &= \left|\mathrm{E}\left[\prod_{j<i} f_j(D_j+E_j)\cdot f_i^H(D_i+E_i)\cdot\prod_{\ell>i} f_\ell^L(D_\ell+E_\ell)\right]\right| \\
&\le \mathop{\mathrm{E}}_D\left[\prod_{j<i}\left|\mathop{\mathrm{E}}_{E_j}[f_j(D_j+E_j)]\right|\cdot\left|\mathop{\mathrm{E}}_{E_i}[f_i^H(D_i+E_i)]\right|\cdot\prod_{\ell>i}\left|\mathop{\mathrm{E}}_{E_\ell}[f_\ell^L(D_\ell+E_\ell)]\right|\right] \\
&\le \mathop{\mathrm{E}}_D\left[\left|\mathop{\mathrm{E}}_{E_i}[f_i^H(D_i+E_i)]\right|\cdot\prod_{\ell>i}\left|\mathop{\mathrm{E}}_{E_\ell}[f_\ell^L(D_\ell+E_\ell)]\right|\right],
\end{aligned}$$

where the last inequality holds because $|\mathrm{E}_{E_j}[f_j(D_j+E_j)]| \le \mathrm{E}_{E_j}[|f_j(D_j+E_j)|] \le 1$ for every $j<i$, by Jensen's inequality, convexity of norms, and the fact that the range of f_j is $\mathbb{C}_1$.

By the Cauchy–Schwarz inequality, we get

$$|\mathrm{E}[F_i(D+E)]| \le \mathop{\mathrm{E}}_D\left[\left|\mathop{\mathrm{E}}_{E_i}[f_i^H(D_i+E_i)]\right|^2\right]^{1/2}\cdot\mathop{\mathrm{E}}_D\left[\prod_{\ell>i}\left|\mathop{\mathrm{E}}_{E_\ell}[f_\ell^L(D_\ell+E_\ell)]\right|^2\right]^{1/2}.$$

In claims 31 and 32 below we bound above the square of the two terms on the right-hand side. In both cases, we view our task as bounding $|\mathrm{E}_D[g(D)]|$ for a certain function g, and

we proceed by computing the L_1-norm, average over uniform, and degree of g, and then we apply Fact 27.

We start with a claim that is useful in both cases.

▶ **Claim 30.** *Let $f\colon [s]^n \to \mathbb{C}$ be a function. Then:*
(1) *for every x, $\mathrm{E}_E[f(x+E)] = \sum_\alpha \hat{f}_\alpha \chi_\alpha(x)(1-\eta)^{|\alpha|}$, and*
(2) $\mathrm{E}_U\left[\left|\mathrm{E}_E[f(U+E)]\right|^2\right] = \sum_\alpha |\hat{f}_\alpha|^2(1-\eta)^{2|\alpha|}$.

Proof. For (1), write $\mathrm{E}_E[f(x+E)] = \mathrm{E}_E[\sum_\alpha \hat{f}_\alpha\chi_\alpha(x+E)] = \sum_\alpha \hat{f}_\alpha\chi_\alpha(x)\,\mathrm{E}_E[\chi_\alpha(E)]$. Then apply Fact 26.

For (2), write $\left|\mathrm{E}_E[f(x+E)]\right|^2$ as $\mathrm{E}_E[f(x+E)]\overline{\mathrm{E}_E[f(x+E)]}$. Then apply (1) twice to further write it as

$$\mathop{\mathrm{E}}_U\left[\sum_{\alpha,\alpha'} \hat{f}_\alpha\overline{\hat{f}_{\alpha'}}\chi_{\alpha-\alpha'}(U)(1-\eta)^{|\alpha|+|\alpha'|}\right] = \sum_{\alpha,\alpha'} \hat{f}_\alpha\overline{\hat{f}_{\alpha'}}\mathop{\mathrm{E}}_U[\chi_{\alpha-\alpha'}(U)](1-\eta)^{|\alpha|+|\alpha'|}.$$

The claim then follows because U is uniform. ◀

We can now bound our terms.

▶ **Claim 31.** *For every i, $\mathrm{E}_D\left[\left|\mathrm{E}_{E_i}[f_i^H(D_i+E_i)]\right|^2\right] \le (1-\eta)^{2t}(1+s^n\delta)$.*

Proof. Let $g(x)$ be the function $g(x) = \mathrm{E}_{E_i}[f_i^H(x+E_i)]$. By (1) in Claim 30, the L_1-norm of g is at most $\sum_{\alpha:|\alpha|>t}|\hat{f}_\alpha|(1-\eta)^{|\alpha|} \le (1-\eta)^t\sum_\alpha|\hat{f}_\alpha| \le (1-\eta)^t s^{n/2}$, where we used Cauchy–Schwarz and Fact 25.

By (2) in Claim 30 and Fact 25, $\mathrm{E}_U[|g(U)|^2]$ under uniform is at most $(1-\eta)^{2t}$.

Because $b \ge n$ we can apply (3) in Fact 27 to obtain that $\mathrm{E}_D[|g(D)|^2] \le (1-\eta)^{2t} + (1-\eta)^{2t}s^n\delta$ as claimed. ◀

▶ **Claim 32.** $\mathrm{E}_D\left[\prod_{\ell>i}\left|\mathrm{E}_{E_\ell}[f_\ell^L(D_\ell+E_\ell)]\right|^2\right] \le 1 + V(t)^{k-1}\delta$.

Proof. Pick any $\ell > i$ and let $g_\ell(x) := \mathrm{E}_E[f_\ell^L(x+E_\ell)]$.

The L_1-norm of g_ℓ is at most $V(t)^{1/2}$ by (1) in Claim 30 and Cauchy–Schwarz. Also by (2) in the same claim we have $\mathrm{E}_U[|g_\ell(U)|^2] \le 1$. Moreover, g_ℓ has degree at most t by (1) in the same claim.

Now define $g\colon ([s]^n)^{k-i} \to \mathbb{C}$ as $g(x_{i+1}, x_{i+2}, \ldots, x_k) := g_{i+1}(x_{i+1})\cdot g_{i+2}(x_{i+2})\cdots g_k(x_k)$. Note that g has L_1-norm at most $V(t)^{(k-i)/2} \le V(t)^{(k-1)/2}$ and degree $(k-i)t \le (k-1)t$, by Fact 24 applied with $u = n(k-i)$. Moreover, $\mathrm{E}_{U_{i+1},U_{i+2},\ldots,U_k}[|g(U_{i+1},U_{i+2},\ldots,U_k)|^2] = \mathrm{E}_{U_{i+1}}[|g_{i+1}|^2]\cdot\mathrm{E}_{U_{i+2}}[|g_{i+2}|^2]\cdots\mathrm{E}_{U_k}[|g_k|^2] \le 1$.

Because $b \ge 2(k-1)t$, we can apply (2) in Fact 27 to obtain

$$\mathop{\mathrm{E}}_D[|g(D)|^2] \le 1 + V(t)^{k-1}\delta$$

as desired. ◀

Lemma 29 follows by combining Claims 31 and 32. ◀

3 Proofs for §1.1

In this section we provide the proofs for the claims made in §1.1.

Proof of Theorem 12. Let L be the set of the 2^c leaves of the protocol tree. For $\ell \in L$, note that the set of inputs that lead to ℓ forms a rectangle, denoted R_ℓ. Moreover, these rectangles are disjoint.

Hence, applying Theorem 5 to each R_ℓ we can write

$$\begin{aligned}|\Pr[P(C+E)=1] - \Pr[P(U)=1]| &= |\sum_\ell \Pr[C+E \in R_\ell] - \sum_\ell \Pr[U \in R_\ell]| \\ &\le \sum_\ell |\Pr[C+E \in R_\ell] - \Pr[U \in R_\ell]|,\end{aligned}$$

from which the result follows. ◀

Recall that we denote by $V(t)$ the number of $x \in \mathbb{F}_q^m$ with at most t non-zero coordinates.

▶ **Claim 33.** *The following two inequalities hold:* $V(t) \le \binom{m}{t} q^t \le (emq/t)^t$.

Proof. The second is standard. To see the first, note that to specify a string with Hamming weight $\le t$ we can specify a super-set of size t of the non-zero positions, and then values for those positions, including 0. ◀

Proof of Theorem 13. Let $H \in \mathbb{F}_q^{(m-d)\times m}$ be the parity-check matrix of C. Let H' be the matrix consisting of the first s rows of H. Note that we do have at least this many rows by our hypothesis on d. Also note that H' has full rank.

On input $x \in \mathbb{F}_q^m$, the algorithm computes $H'x$, and accepts if and only if $H'x$ equals to $H'e$ for any $e \in \mathbb{F}_q^m$ of Hamming weight at most t.

To analyze the correctness, let y be a codeword with at most t errors. Then $H(y-e)=0$ for some $e \in \mathbb{F}_q^m$ with Hamming weight at most t, and so the algorithm always accepts. On the other hand if U is uniform, then as H' has full rank, $H'U$ is uniform in $\mathbb{F}_q^s$. Since there are $V(t)$ vectors in $\mathbb{F}_q^m$ with Hamming weight at most t, the algorithm accepts with probability $\le V(t)/q^s \le \delta$.

Now we show how to compute $H'x$ using s symbols of space (and so $s \log q$ bits). For $i \le s$, let h_i be the i-th row of H'. Note that the i-th symbol of $H'x$ equals $\sum_{j \le n} h_{i,j} x_j$, which can be computed with one symbol of space by keeping the partial sum. The result follows.

The "moreover" part follows from Claim 33. ◀

Proof of Theorem 14. Suppose

$$\Pr\left[P(GU+E) = U_i\right] \ge 1/q + \varepsilon.$$

Let D_a be the uniform distribution over $\{Gx \mid x_i = a\}$. We can rewrite the inequality as

$$\mathop{\mathrm{E}}_{a \in \mathbb{F}_q}[\Pr[P(D_a+E)=a] - \Pr[P(U)=a]] \ge \varepsilon.$$

Therefore, there exists an a such that $\Pr[P(D_a+E)=a] - \Pr[P(U)=a] \ge \varepsilon$.

We now use P to construct a protocol P' that distinguishes D_0+E from uniform. Given $y \in \mathbb{F}_q^m$, the parties add to y the ith column G_i of G multiplied by a. This can be done

without communication. Then they run the protocol P on $y + aG_i$ and accept if and only if the output is a. We have

$$\begin{aligned}\Pr[P'(D_0 + E) = 1] - \Pr[P'(U) = 1] &= \Pr[P(D_0 + aG_i + E) = a] - \Pr[P(U) = a] \\ &= \Pr[P(D_a + E) = a] - \Pr[P(U) = a] \\ &\geq \varepsilon.\end{aligned}$$

So the result follows from Theorem 12. ◀

Proof of Theorem 15. Let $n := q/k$ be the input length to a party. The parties communicate $m - d + 2t + 1 - n$ symbols that the first does not have, and no symbol if $m - d + 2t + 1 - n \leq 0$. The first party then outputs the unique message whose encoding is at distance $\leq t$ with the $m - d + 2t + 1$ symbols z they have, i.e., the symbols they received plus the n they already have. The message corresponding to x clearly is such a message. Also no other such message exists, because if two encodings are at distance $\leq t$ with z then they agree with each other in $\geq m - d + 1$ symbols, and so they cannot differ in d positions and must be the same. ◀

4 Pseudorandomness: I

In this section we prove our first theorem on pseudorandom generators, Theorem 18.

First, we shall need the following lemma to sample our noise vectors, which is also used in the next section. We write SD for *statistical distance*.

▶ **Lemma 34.** *There is a polynomial-time computable function f mapping $O(\eta \log(1/\eta)m)$ bits to $\{0,1\}^m$ such that $\mathrm{SD}(f(U), E) \leq e^{-\Omega(\eta m)}$.*

In turn, that will use the following lemma to sample arbitrary distributions through discretization. A version of the lemma appears in [51], Lemma 5.2. That version only bounds the number of bits of the sampler. Here we also need that the sampler is efficient.

▶ **Lemma 35.** *Let D be a distribution on $S := \{1, 2, \ldots, n\}$. Suppose that given $i \in S$ we can compute in time polynomial in $|i| = O(\log n)$ the cumulative distribution $\Pr[D \leq i]$.*

Then there is a polynomial-time computable function f such that given any $t \geq 1$ uses $\lceil \log_2 nt \rceil$ bits to sample a string in the support of D such that $\mathrm{SD}(f(U), D) \leq 1/t$.

Proof. Following [51, Lemma 5.2], partition the interval $[0,1]$ into n intervals I_i of lengths $\Pr[D = i]$, $i = 1, \ldots, n$. Also partition $[0,1]$ in $\ell := 2^{\lceil \log_2 nt \rceil} \geq nt$ intervals of size $1/\ell$ each, which we call blocks. The function f interprets an input as a choice of a block b, and outputs i if $b \subseteq I_i$ and, say, outputs 1 if b is not contained in any interval.

For any i we have $|\Pr[D = i] - \Pr[f(U) = i]| \leq 2/\ell$. Hence the statistical distance is $\leq (1/2) \sum_i |\Pr[D = i] - \Pr[f(U) = i]| \leq (1/2) n 2/\ell \leq 1/t$.

To show efficiency we have to explain how given b we determine the i such that $b \subseteq I_i$. We perform binary search. This requires $O(\log n)$ steps, and in each step we compute the cumulative distribution function of D, which by assumption is in polynomial time. ◀

Proof of Lemma 34. Our function f first samples a weight distribution W on $\{0, \ldots, m\}$ so that $\mathrm{SD}(W, |E|) \leq e^{-\Omega(\eta m)}$. By Lemma 35, this uses a seed of length $O(\eta m + \log(m+1))$ and runs in polynomial time. Given a sample $w \sim W$. If $w \geq 2\eta m$, we output the all-zero string. Otherwise we sample a string in $\{0,1\}^m$ with Hamming weight w almost uniformly. To do this, first we index the $\binom{m}{w}$ strings in lexicographical order. We then use Lemma 35 again to sample an index in $\{1, \ldots, \binom{m}{w}\}$ from a distribution that is $e^{-\Omega(\eta m)}$-close to uniform.

This takes another seed of length at most $O(\eta m + \log\binom{m}{2\eta m}) = O(\eta m + \eta\log(1/\eta)m)$ and can be computed in polynomial time.

Given an index i, we output the corresponding string efficiently using the following recurrence. Let $s(m,k,i)$ denote the i-th m-bit string with Hamming weight k, in lexicographical order. We have

$$s(m,k,i) = \begin{cases} 0 \circ s(m-1,k,i) & \text{if } i \le \binom{m-1}{k} \\ 1 \circ s(m-1,k-1,i-\binom{m-1}{k}) & \text{otherwise.} \end{cases}$$

Note that $s(m,k,i)$ outputs the string by m comparisons of $\lceil\binom{m}{2\eta m}\rceil$-bit strings, and thus can be computed in polynomial time.

Therefore f has input length $O(\eta m + \eta\log(1/\eta)m + \log(m+1)) = O(\eta\log(1/\eta)m)$. Let $D := f(U)$. We now bound above the statistical distance between D and E. Denote D_w as the distribution D conditioned on $|D| = w$ and denote E_w analoguously. We have

$$\begin{aligned}\sum_{x\in\{0,1\}^m} |\Pr[D(x)] - \Pr[E(x)]| &= \sum_{w=0}^{m}\sum_{|x|=w} |\Pr[D(x)] - \Pr[E(x)]| \\ &= \sum_{w=0}^{m}\sum_{|x|=w} |\Pr[D_w(x)]\Pr[|D|=w] - \Pr[E_w(x)]\Pr[|E|=w]|,\end{aligned}$$

Adding $-\Pr[D_w(x)]\Pr[|E|=w] + \Pr[D_w(x)]\Pr[|E|=w] = 0$ in each summand, this is at most

$$\sum_{w=0}^{m}\sum_{|x|=w} \Pr[D_w(x)]\cdot|\Pr[|D|=w]-\Pr[|E|=w]| + \sum_{w=0}^{m}\sum_{|x|=w} |\Pr[D_w(x)]-\Pr[E_w(x)]|\cdot\Pr[|E|=w].$$

The first double summation is at most $2\,\mathrm{SD}(|D|,|E|) = 2\,\mathrm{SD}(W,|E|)$. We now bound above the second summation as follows. We separate the outer sum into $w > 2\eta m$ and $w \le 2\eta m$. For the first case, we have

$$\sum_{w>2\eta m}\sum_{|x|=w} |\Pr[D_w(x)] - \Pr[E_w(x)]| \cdot \Pr[|E|=w] \le 2\Pr[|E| > 2\eta m].$$

By the Chernoff Bound, this is at most $2e^{-\Omega(\eta m)}$. For the other case, we have

$$\sum_{w\le 2\eta m}\sum_{|x|=w} |\Pr[D_w(x)] - \Pr[E_w(x)]| \cdot \Pr[|E|=w] \le 2\max_{w\le 2\eta m}\mathrm{SD}(D_w,E_w).$$

Therefore,

$$\mathrm{SD}(D,E) \le \mathrm{SD}(W,|E|) + \max_{w\le 2\eta m}\mathrm{SD}(D_w,E_w) + e^{-\Omega(\eta m)} \le 3e^{-\Omega(\eta m)}. \qquad \blacktriangleleft$$

We can now prove our first theorem on pseudorandom generators.

Proof of Theorem 18. (1) We apply Theorem 23. Known constructions [6, Theorem 2] (see also [41]) produce a δ-biased distribution over m bits using $2\log(1/\delta) + O(\log m)$ bits. We set $\delta = O(2^{-n}\varepsilon)$, resulting in a seed length of $2n + 2\log(1/\varepsilon) + O(\log m)$ bits.

For the noise we set $\eta = O(k\log k\log(k/\varepsilon)/n)$. Note that $\eta \le 1$ because we can assume $k^2\log k\log(k/\varepsilon)\log n \le n$, for else (2) gives a better bound.

By Lemma 34, the seed length to generate the noise vector is $O(k^2\log k\log(k/\varepsilon)\log(n/k))$.

In Theorem 23 set $t = cn/(k \log k)$ for a small enough constant c. Then we can bound $V(t)^{k/2} \leq V(t)^{k-1} \leq 2^n$. Thus the error bound from Theorem 23 is at most $k(1-\eta)^{cn/(k \log k)}(1 + 2^n\delta) + 2^n\delta \leq 2k(1-\eta)^{cn/(k \log k)} + \varepsilon/4 \leq \varepsilon/2$.

The error from Lemma 34 is $e^{-\Omega(\eta m)} \leq \varepsilon/2$. Thus overall the error is at most ε.

The fact that we can apply any permutation π follows from the fact that applying such a permutation does not change the noise distribution, and preserves the property of being b-wise independent.

(2) Let $c := \lfloor (k^2 \log k \log(k/\varepsilon) \log n/n)^{1/3} \rfloor$. We can assume $c \geq 1$ for else (1) gives a better bound. Group the k functions into $k' = \lceil k/c \rceil$ functions on input length $n' := cn$. We can now apply (1) to n' and k' to get the desired seed length. ◀

5 Pseudorandomness, II

We now move to our second theorem on pseudorandom generators, Theorem 19. We begin by modifying Theorem 23 to allow us to sample the noise in a certain pseudorandom way. Specifically, we can write our noise vector E in the previous sections as $E = T \wedge U$, where U is uniform, T is a distribution of i.i.d. bits where each comes 1 with probability η, and $\wedge$ denotes bit-wise And. In the derandomized way, we keep U uniform but select T using an almost n-wise independent distribution. The analogue of Theorem 23 with this derandomization is proved below as Theorem 37. Finally, we show how to recurse on U in §5.2.

At the end of the section we show that a certain improvement in the error bound of Theorem 37 would yield much better pseudorandom generators.

▶ **Definition 36.** A distribution T on m bits is *γ-almost d-wise independent* if for every d indices $i_1, \ldots, i_d$ and any $S \subseteq \{0,1\}^d$ we have

$$\left| \sum_{x \in S} \left(\Pr[\bigwedge_{j \leq d} T_{i_j} = x_j] - \prod_{j \leq d} \Pr[T_{i_j} = x_j] \right) \right| \leq \gamma.$$

▶ **Theorem 37** (Bounded Independence Plus Derandomized Noise Fools Products)**.** *Let $t \in [0,n]$. Let $f_1, \ldots, f_k \colon \{0,1\}^n \to \mathbb{C}_1$ be k functions with $\mu_i = \mathrm{E}[f_i]$. Let D be an δ-biased distribution over $(\{0,1\}^n)^k$. Let T be a γ-almost n-wise distribution over $(\{0,1\}^n)^k$ which sets each bit to 1 with probability η and 0 otherwise. Assume $\gamma \leq \eta$. Let U be the uniform distribution over $(\{0,1\}^n)^k$. Write $D = (D_1, D_2, \ldots, D_k)$ where each D_i is in $\{0,1\}^n$, and similarly for T and U. Then*

$$\left| \mathrm{E}\left[\prod_{i \leq k} f_i(D_i + T_i \wedge U_i) \right] - \prod_{i \leq k} \mu_i \right| \leq k((1-\eta)^t + \gamma)^{1/2} \sqrt{(1 + 2^n\delta)(1 + V(t)^{k-1}\delta)} + V(t)^{k/2}\delta.$$

5.1 Proof of Theorem 37

We begin exactly as in the proof of Theorem 23. For a function $f \colon \{0,1\}^n \to \mathbb{C}_1$, consider its Fourier expansion $f(x) := \sum_\alpha \hat{f}_\alpha \chi_\alpha(x)$, and let $f^L(x) := \sum_{\alpha : |\alpha| \leq t} \hat{f}_\alpha \chi_\alpha(x)$ and $f^H(x) := \sum_{\alpha : |\alpha| > t} \hat{f}_\alpha \chi_\alpha(x)$. Define $F_i \colon (\{0,1\}^n)^k \to \mathbb{C}$ to be

$$F_i(x_1, \ldots, x_k) := \left(\prod_{j<i} f_j(x_j) \right) \cdot f_i^H(x_i) \cdot \left(\prod_{\ell > i} f_\ell^L(x_\ell) \right).$$

Pick f_k and write it as $f_k^L + f_k^H$. We can then rewrite

$$\prod_{1\le i\le k} f_i = F_k + \left(\prod_{1\le i\le k-1} f_i\right)\cdot f_k^L.$$

We can reapply the process to $(\prod_{1\le i\le k-1} f_i)$. Continuing this way, we eventually have what we want to bound, i.e. $|\mathrm{E}[\prod_{i\le k} f_i(D_i+T_i\wedge U_i)] - \prod_{i\le k}\mu_i|$, is at most

$$\left|\sum_{i\le k}\mathrm{E}[F_i(D+T\wedge U)]\right| + \left|\mathrm{E}[\prod_{i\le k} f_i^L(D_i+T_i\wedge U_i)] - \prod_{i\le k}\mu_i\right|.$$

The theorem follows readily from the next two lemmas, the second of which has a longer proof. The first one has the same proof as Lemma 28.

▶ **Lemma 38.** $|\mathrm{E}_{D,T,U}[\prod_{i\le k} f_i^L(D_i+T_i\wedge U_i)] - \prod_{i\le k}\mu_i| \le V(t)^{k/2}\delta$.

▶ **Lemma 39.** *For every $i\in\{1,2,\dots,k\}$, we have*

$$|\mathop{\mathrm{E}}_{D,T,U}[F_i(D+T\wedge U)]| \le ((1-\eta)^t+\gamma)^{1/2}\sqrt{(1+2^n\delta)(1+V(t)^{k-1}\delta)}.$$

Proof. We have

$$\begin{aligned}
&|\mathrm{E}[F_i(D+T\wedge U)]|\\
&= \left|\mathrm{E}\left[\prod_{j<i} f_j(D_j+T_j\wedge U_j)\cdot f_i^H(D_i+T_i\wedge U_i)\cdot\prod_{\ell>i} f_\ell^L(D_\ell+T_\ell\wedge U_\ell)\right]\right|\\
&\le \mathop{\mathrm{E}}_{D,T}\left[\prod_{j<i}\left|\mathop{\mathrm{E}}_{U_j}[f_j(D_j+T_j\wedge U_j)]\right|\cdot\left|\mathop{\mathrm{E}}_{U_i}[f_i^H(D_i+T_i\wedge U_i)]\right|\cdot\prod_{\ell>i}\left|\mathop{\mathrm{E}}_{U_\ell}[f_\ell^L(D_\ell+T_\ell\wedge U_\ell)]\right|\right]\\
&\le \mathop{\mathrm{E}}_{D,T}\left[\left|\mathop{\mathrm{E}}_{U_i}[f_i^H(D_i+T_i\wedge U_i)]\right|\cdot\prod_{\ell>i}\left|\mathop{\mathrm{E}}_{U_\ell}[f_\ell^L(D_\ell+T_\ell\wedge U_\ell)]\right|\right],
\end{aligned}$$

where the last inequality holds because $|\mathrm{E}_{U_j}[f_j(D_j+T_j\wedge U_j)]| \le \mathrm{E}_{U_j}[|f_j(D_j+T_j\wedge U_j)|]\le 1$ for every $j<i$, by Jensen's inequality, convexity of norms, and the fact that the range of f_j is $\mathbb{C}_1$.

By the Cauchy–Schwarz inequality, we get

$$|\mathrm{E}[F_i(D+T\wedge U)]| \le \mathop{\mathrm{E}}_{D,T}\left[\left|\mathop{\mathrm{E}}_{U_i}[f_i^H(D_i+T_i\wedge U_i)]\right|^2\right]^{1/2}\cdot\mathop{\mathrm{E}}_{D,T}\left[\prod_{\ell>i}\left|\mathop{\mathrm{E}}_{U_\ell}[f_\ell^L(D_\ell+T_\ell\wedge U_\ell)]\right|^2\right]^{1/2}.$$

In claims 41 and 42 below we bound from above the square of the two terms on the right-hand side. In both cases, we view our task as bounding $|\mathrm{E}_D[g(D)]|$ for a certain function g, and we proceed by computing the L_1-norm, average over uniform, and degree of g, and then we apply Fact 27.

We start with a claim that is useful in both cases.

▶ **Claim 40** (Replacing Claim 30). *Let $f\colon\{0,1\}^n\to\mathbb{C}$ be a function. Let T be a γ-almost n-wise independent distribution which sets each bit to 1 with probability η and 0 otherwise. Let U and U' be two independent uniform distributions over n bits. Then:*
(1) *for every x, $\mathrm{E}_{T,U}[f(x+T\wedge U)] = \sum_\alpha \hat{f}_\alpha\chi_\alpha(x)((1-\eta)^{|\alpha|}+\gamma)$, and*
(2) $\mathrm{E}_{U,T}\left[|\mathrm{E}_{U'}[f(U+T\wedge U')]|^2\right] = \sum_\alpha|\hat{f}_\alpha|^2((1-\eta)^{|\alpha|}+\gamma)$.

Proof. For (1), write

$$\mathop{\mathrm{E}}_{T,U}[f(x+T\wedge U)] = \mathop{\mathrm{E}}_{T,U}[\sum_{\alpha}\hat{f}_\alpha\chi_\alpha(x+T\wedge U)] = \sum_{\alpha}\hat{f}_\alpha\chi_\alpha(x)\mathop{\mathrm{E}}_{T,U}[\chi_\alpha(T\wedge U)].$$

If T does not intersect α then the expectation is one, and this happens with probability at most $(1-\eta)^{|\alpha|}+\gamma$. Otherwise, the expectation is 0.

For (2), write $\mathrm{E}_{U,T}[|\mathrm{E}_{U'}[f(x+T\wedge U')]|^2]$ as

$$\mathop{\mathrm{E}}_{U,T}\left[\sum_{\alpha,\alpha'}\hat{f}_\alpha\overline{\hat{f}_{\alpha'}}\chi_{\alpha-\alpha'}(U)\mathop{\mathrm{E}}_{U',U''}[\chi_\alpha(T\wedge U')\overline{\chi_{\alpha'}(T\wedge U'')}]\right].$$

Since U is uniform this becomes $\sum_\alpha|\hat{f}_\alpha|^2\,\mathrm{E}_{T,U',U''}[\chi_\alpha(T\wedge(U'-U''))] = \sum_\alpha|f_\alpha|^2\,\mathrm{E}[\chi_\alpha(T\wedge U)]$. The claim then follows as in (1). ◀

We can now bound our terms.

▶ **Claim 41** (Replacing Claim 31). *For every i, $\mathrm{E}_{D,T}[|\mathrm{E}_U[f_i^H(D_i+T_i\wedge U_i)]|^2] \le ((1-\eta)^t+\gamma)(1+2^n\delta)$.*

Proof. Let $g(x)$ be the function $g(x)=\mathrm{E}_{T_i,U_i}[f_i^H(x+T_i\wedge U_i)]$. By (1) in Claim 40, the L_1-norm of g is at most $\sum_{\alpha:|\alpha|>t}|\hat{f}_\alpha|((1-\eta)^{|\alpha|}+\gamma) \le ((1-\eta)^t+\gamma)\sum_\alpha|\hat{f}_\alpha| \le ((1-\eta)^t+\gamma)2^{n/2}$, where we used Cauchy–Schwarz and Fact 25.

Also, by (2) in Claim 40 and Fact 25, $\mathrm{E}_U[|g(U)|^2]$ under uniform is at most $(1-\eta)^t+\gamma$.

Because $nk \ge n$ we can apply (3) in Fact 27 to obtain that $\mathrm{E}_D[|g(D)|^2] \le ((1-\eta)^t+\gamma)+((1-\eta)^t+\gamma)^2 2^n\delta \le ((1-\eta)^t+\gamma)(1+2^n\delta)$ as claimed. ◀

▶ **Claim 42.** $\mathrm{E}_{D,T}\left[\prod_{\ell>i}\left|\mathrm{E}_{U_\ell}[f_\ell^L(D_\ell+T_\ell\wedge U_\ell)]\right|^2\right] \le 1+V(t)^{k-1}\delta.$

Proof. Pick any $\ell>i$ and let $g_\ell(x) := \mathrm{E}_E[f_\ell^L(x+E_\ell)]$.

The L_1-norm of g_ℓ is at most $V(t)^{1/2}$ by (1) in Claim 30 and Cauchy–Schwarz. Also by (2) in the same claim we have $\mathrm{E}_U[|g_\ell(U)|^2] \le 1$. Moreover, g_ℓ has degree at most t by (1) in the same claim.

Now define $g\colon([s]^n)^{k-i}\to\mathbb{C}$ as $g(x_{i+1},x_{i+2},\ldots,x_k) := g_{i+1}(x_{i+1})\cdot g_{i+2}(x_{i+2})\cdots g_k(x_k)$. Note that g has L_1-norm at most $V(t)^{(k-i)/2} \le V(t)^{(k-1)/2}$ and degree $(k-i)t \le (k-1)t$, by Fact 24 applied with $u=n(k-i)$. Moreover, $\mathrm{E}_{U_{i+1},U_{i+2},\ldots,U_k}[|g(U_{i+1},U_{i+2},\ldots,U_k)|^2] = \mathrm{E}_{U_{i+1}}[|g_{i+1}|^2]\cdot\mathrm{E}_{U_{i+2}}[|g_{i+2}|^2]\cdots\mathrm{E}_{U_k}[|g_k|^2] \le 1$.

Because $b \ge 2(k-1)t$, we can apply (2) in Fact 27 to obtain

$$\mathop{\mathrm{E}}_{D}[|g(D)|^2] \le 1+V(t)^{k-1}\delta$$

as desired. ◀

Proof. Pick any $\ell>i$ and let $g_\ell(x) := \mathrm{E}_{T,U_\ell}[f_\ell^L(x+T_\ell\wedge U_\ell)]$.

The L_1-norm of g_ℓ is at most $V(t)^{1/2}$ by (1) in Claim 40, Cauchy–Schwarz, and the assumption that $\gamma\le\eta$. Also by (2) in the same claim we have $\mathrm{E}_U[|g_\ell(U)|^2] \le 1$. Moreover, g_ℓ has degree at most t by (1) in the same claim.

Now define $g\colon([s]^n)^{k-i}\to\mathbb{C}$ as $g(x_{i+1},x_{i+2},\ldots,x_k) := g_{i+1}(x_{i+1})\cdot g_{i+2}(x_{i+2})\cdots g_k(x_k)$. Note that g has L_1-norm at most $V(t)^{(k-i)/2} \le V(t)^{(k-1)/2}$ and degree $(k-i)t \le (k-1)t$, by Fact 24 applied with $u=n(k-i)$. Moreover, $\mathrm{E}_{U_{i+1},U_{i+2},\ldots,U_k}[|g(U_{i+1},U_{i+2},\ldots,U_k)|^2] = \mathrm{E}_{U_{i+1}}[|g_{i+1}|^2]\cdot\mathrm{E}_{U_{i+2}}[|g_{i+2}|^2]\cdots\mathrm{E}_{U_k}[|g_k|^2] \le 1$.

Because $nk \geq 2(k-1)t$, we can apply (2) in Fact 27 to obtain

$$\mathop{\mathrm{E}}_{D}[|g(D)|^2] \leq 1 + V(t)^{k-1}\delta$$

as desired. ◀

Lemma 39 follows by combining Claims 41 and 42. ◀

5.2 A recursive generator

▶ **Lemma 43.** *Suppose $n \geq Ck \log k \log(k/\varepsilon)$ for a universal constant C. Let c be an integer. If there is an explicit generator $G_{cn/4,k/c}\colon \{0,1\}^{\ell} \to (\{0,1\}^{cn/4})^{k/c}$ that ε-fools product tests that read bits in any order and uses a seed of length ℓ, then there is an explicit generator $G_{n,k}\colon \{0,1\}^{\ell'} \to (\{0,1\}^n)^k$ that fools product tests in any order with error $\varepsilon/k + \varepsilon$ and uses a seed of length $\ell' = O(n) + \ell$.*

Proof. Our generator $G_{n,k}\colon \{0,1\}^{\ell'} \to (\{0,1\}^n)^k$ samples a 2^{-2n}-biased distribution D on m bits, and a 2^{-2n}-almost n-wise independent distribution $T = (T_1, \ldots, T_k)$ on m bits which sets each bit to 1 with probability $1/8$ and 0 otherwise. If $|T_i| > n/4$ for some $1 \leq i \leq k$, let G output the all-zero m-bit string. Otherwise, output $D + T \wedge \mathrm{PAD}_T(G_{cn/4,k/c}(U_\ell))$, where $\mathrm{PAD}_T(x)_j$ is defined as follows: If $j \in T$, $\mathrm{PAD}_T(x)_j$ equals the first bit of x that has not appeared in the first $j-1$ bits of $\mathrm{PAD}_T(x)$. Otherwise $\mathrm{PAD}_T(x)_j = 0$. Note that $|T| \leq m/4$.

Now we analyze the seed length of $G_{n,k}$. Standard constructions [41, 6] use a seed of $O(n)$ bits to sample D. To sample T, we will use the following lemma from [45].

▶ **Lemma 44** (Lemma B.2 in [45])**.** *There is an explicit sampler that samples a γ-almost n-wise independent distribution T on m bits which sets each bit to 1 with probability η and 0 otherwise and uses a seed of length $O(n \log(1/\eta) + \log((\log m)/\gamma))$.*

Applying the lemma with $\gamma = 2^{-2n}$ and $\eta = 1/8$, we can sample T with $O(n)$ bits. So the total seed length of $G_{n,k}$ is $\ell' = O(n) + \ell$.

We now analyze the error of $G_{n,k}$. Let $f\colon (\{0,1\}^n)^k \to \mathbb{C}_1$ be a product test. We bound above $|\mathrm{E}[f(U_m)] - \mathrm{E}[f(G_{n,k}(U_{\ell'}))]|$ by

$$|\mathrm{E}[f(U_m)] - \mathrm{E}[f(D + T \wedge U_m)]| + |\mathrm{E}[f(D + T \wedge U_m)] - \mathrm{E}[f(G_{n,k}(U_{\ell'})]|.$$

The first term is at most $\varepsilon/2k$ by Theorem 37 with the following choice of parameters. We set $t = cn/(k \log k)$ for a small enough constant c. Then we can bound $V(t)^{k/2} \leq V(t)^{k-1} \leq 2^n$. We set $\eta = 1/8$. Thus, by the condition $n \geq Ck \log k \log(k/\varepsilon)$ the error bound from Theorem 37 is at most $k((1-\eta)^{cn/(k \log k)} + \gamma)^{1/2}(1 + 2^n\delta) + 2^n\delta \leq O(k((\varepsilon/k)^{100} + 2^{-2n})^{1/2}) + 2^{-n} \leq \varepsilon/2k$.

For the second term, let T' be T conditioned on $|T_i| \leq n/4$ for every $1 \leq i \leq k$. For every fixed $y \in D$ and $t \in T'$, consider the function $f_{y,t}\colon (\{0,1\}^{n/4})^k \to \mathbb{C}_1$ by $f_{y,t}(x) := f(y + t \wedge \mathrm{PAD}_t(x))$. Note that we can group every c functions into one and think of $f_{y,t}$ as a product test of k/c functions on $cn/4$ bits, which can be fooled by $G_{cn/4,k/c}$. Thus, $|\mathrm{E}[f(D + T' \wedge U_m)] - \mathrm{E}[f(G_{n,k}(U_{\ell'}))]|$ equals

$$\begin{aligned}
&|\mathrm{E}[f(D + T' \wedge \mathrm{PAD}_{T'}(U_{m/4}))] - \mathrm{E}[f(D + T' \wedge \mathrm{PAD}_{T'}(G_{cn/4,k/c}(U_\ell)))]| \\
&\leq \mathop{\mathrm{E}}_{y \sim D, t \sim T'}[|\mathrm{E}[f_{y,t}(U_{m/4})] - \mathrm{E}[f_{y,t}(G_{cn/4,k/c}(U_\ell))]|] \\
&\leq \varepsilon.
\end{aligned}$$

Now let E denote the event $|T_i| > n/4$ for some $1 \leq i \leq k$. We bound above $\Pr[E]$. We will use the following tail bound for almost d-wise independence.

▶ **Lemma 45** (Lemma B.1 in [45]). *Let $T = (T_1, \ldots, T_n)$ be a γ-almost d-wise independent distribution on n bits where $\mathrm{E}[T_j] = \eta$ for every $1 \le j \le n$. Then for any $\varepsilon \in (0,1)$,*

$$\Pr[|\sum_{i \le n} X_i - \eta n| \ge \varepsilon n] \le \left(\frac{ed}{2\varepsilon^2 n}\right)^{d/2} + \gamma/\varepsilon^d.$$

We apply Lemma 45 with $\eta = \varepsilon = 1/8$, $\gamma = 2^{-2n}$, and $d = \Omega(n)$. This guarantees that for each $1 \le i \le k$ the probability of $|T_i| \le n/4$ is at most $2^{-\Omega(n)}$. By a union bound over $T_1, \ldots, T_k$, we have $\Pr[E] \le k2^{-\Omega(n)} \le \varepsilon/2k$.

Putting everything together we have error $\varepsilon/k + \varepsilon$. ◀

Finally, we combine these results to prove Theorem 19.

Proof of Theorem 19. Let C be the universal constant in Lemma 43. Suppose $n \ge Ck \log k \log(k/\varepsilon)$. We will first apply Lemma 43 with $c = 2$ for $t := O(\log k)$ times until we are left with a product test of $O(1)$ functions on $O(n/k)$ bits, and then we output the uniform $O(n/k)$-bit string. Note that the condition $n \ge Ck \log k \log(k/\varepsilon)$ holds throughout because n and k are both divided by 2 at each step.

Note that in each application of Lemma 43, we reduce n by at least a half. Hence, the total seed length is at most $\sum_{i=0}^{t} O(n/2^i) + O(n/k) = O(n)$. The error is at most $\sum_{i=0}^{t} 2^i\varepsilon/k \le \varepsilon$.

If $n \le Ck \log k \log(k/\varepsilon)$, pick an integer $c = O(\sqrt{k \log k \log(k/\varepsilon)/n})$ so that $c^2 n \ge O(Ck \log k \log(k/\varepsilon))$. By grouping every c functions into one, f is also a product test of k/c functions on cn bits. Hence, by the previous result we have a generator with seed length $\ell = O(cn) = O(\sqrt{nk \log k \log(k/\varepsilon)})$. ◀

A potential improvement. We now show that an improvement in the error bound of Theorem 37 would yield much better pseudorandom generators.

▶ **Claim 46.** *Let D be an n-wise independent distribution on $m := nk$ bits. Let T be an n-wise independent distribution on m bits which sets each bit to 1 with probability η. Let U be the uniform distribution on m bits.*

Suppose that for any product test $f\colon (\{0,1\}^n)^k \to \mathbb{C}_1$ on m bits we have $|\mathrm{E}[f(U)] - \mathrm{E}[f(D + T \wedge U)]| \le k(1-\eta)^{\Omega(n)}$.

Then there is an explicit generator $G\colon \{0,1\}^\ell \to (\{0,1\}^n)^k$ that ε-fools product tests in any order with seed length $\ell = O((n + \log k \log(m/\varepsilon)) \log m)$.

To prove Claim 46, first we replace Lemma 43 with the following lemma.

▶ **Lemma 47.** *Suppose $n \ge C \log(m/\varepsilon)$ for a universal constant C. Let c be an integer. If there is an explicit generator $G_{cn/4,k/c}$ that fools product tests that read bits in any order on $(cn/4) \cdot (k/c)$ bits with error ε and uses a seed of length ℓ, then there is an explicit generator $G_{n,k}\colon \{0,1\}^{\ell'} \to (\{0,1\}^n)^k$ that fools product tests that read bits in any order on $m := nk$ bits with error $\varepsilon/m + \varepsilon$ and uses a seed of $\ell' = O(n \log m) + \ell$ bits.*

Proof. The generator is very similar to the one in Lemma 43 except that G now samples an n-wise independent distribution D on m bits and an n-wise independent distribution T on m bits that sets each bit to 1 with probability $1/8$ and 0 otherwise. Now sampling D and T takes a seed of length $O(n \log m)$ [20, 5].

Now we analyze the error of $G_{n,k}$. Let $f\colon (\{0,1\}^n)^k \to \mathbb{C}_1$ be a product test. As in the proof of Lemma 43 we bound above $|\mathrm{E}[f(U_m)] - \mathrm{E}[f(G_{n,k}(U_{\ell'}))]|$ by

$$|\mathrm{E}[f(U_m)] - \mathrm{E}[f(D + T \wedge U_m)]| + |\mathrm{E}[f(D + T \wedge U_m)] - \mathrm{E}[f(G_{n,k}(U_{\ell'}))]|.$$

By our assumption, the first term is at most $k(1-\eta)^{\Omega(n)} \le \varepsilon/2m$. For the second term, let T' be T conditioned on $|T_i| \le n/4$ for every $1 \le i \le k$. For every fixed $y \in D$ and $t \in T'$, consider the function $f_{y,t}\colon (\{0,1\}^{n/4})^k \to \mathbb{C}_1$ by $f_{y,t}(x) := f(y + t \wedge \mathrm{PAD}_t(x))$. Note that we can group every c functions into one and think of $f_{y,t}$ as a product test of k/c functions on $cn/4$ bits, which can be fooled by $G_{cn/4,k/c}$. Thus, $|\mathrm{E}[f(D + T' \wedge U_m)] - \mathrm{E}[f(G_{n,k}(U_{\ell'}))]|$ equals

$$\begin{aligned}
&|\mathrm{E}[f(D + T' \wedge \mathrm{PAD}_{T'}(U_{m/4}))] - \mathrm{E}[f(D + T' \wedge \mathrm{PAD}_{T'}(G_{cn/4,k/c}(U_\ell)))]| \\
&\le \mathop{\mathrm{E}}_{y \sim D, t \sim T'}[|\mathrm{E}[f_{y,t}(U_{m/4})] - \mathrm{E}[f_{y,t}(G_{cn/4,k/c}(U_\ell))]|] \\
&\le \varepsilon.
\end{aligned}$$

Now let E denote the event $|T_i| > n/4$ for some $1 \le i \le k$. We bound above $\Pr[E]$. Since T is n-wise independent, by the Chernoff bound the probability of $|T_i| \le n/4$ is at most $2^{-\Omega(n)}$. By a union bound over $T_1, \dots, T_k$, we have $\Pr[E] \le k2^{-\Omega(n)} \le \varepsilon/2m$.

Putting everything together we have error $\varepsilon/m + \varepsilon$. ◀

Proof of Claim 46. Suppose $n \ge C\log(m/\varepsilon)$. We apply Lemma 47 recursively, in two different ways. One way reduces n and the other reduces k. First, we apply the lemma with $c = 1$ for $t_1 := O(\log n)$ times to bring n down to $n' = O(\log(m/\varepsilon))$. This takes a seed of $\ell_1 := \sum_{i=0}^{t} O(n \log m/4^i) = O(n \log m)$ bits. Now we have a product test of k functions on n' bits. We will instead think of it as a product test of $k/2$ functions on $2n'$ bits, and apply Lemma 47 with $c = 2$, which will reduce it to a product test of $k/4$ functions on n' bits. Now we repeat $t_2 := O(\log k)$ steps to reduce k to $k' = O(1)$. This takes a seed of $\ell_2 := t_2 \cdot O(n' \log m) = O(\log k \log(m/\varepsilon) \log m)$ bits. Now we are left with a product test of k' functions on n' bits, and we can output the uniform string. Therefore the total seed length is $\ell = \ell_1 + \ell_2 + O(\log(m/\varepsilon)) = O((n + \log k \log(m/\varepsilon)) \log m)$. Because in each application of Lemma 47 the input length of the product test decreases by at least half, the error bound is at most $\sum_{i=0}^{t_1+t_2} 2^i\varepsilon/m \le 2^{O(\log m)}\varepsilon/m \le \varepsilon$.

If $n \le C\log(m/\varepsilon)$, we can group the functions and have a product test of k' functions on $C\log(m/\varepsilon)$ bits where $k' \le k$, and reason as before. ◀

6 Pseudorandomness, III

In this section we prove Theorem 22, giving generators for generalized halfspaces and combinatorial shapes. After that, we discuss the relationship between the results in this paper and an original motivation [38].

▶ **Lemma 48** ([27]). *Suppose $G\colon \{0,1\}^\ell \to (\{0,1\}^n)^k$ is an explicit generator that ε-fools any product test on nk bits that reads bits in any order, then*

1. *G fools any generalized halfspace $h\colon (\{0,1\}^n)^k \to \{0,1\}$ on nk bits that reads bits in any order with error $O(k2^n(n + \log k)\varepsilon)$.*
2. *G fools any combinatorial shape $g\colon (\{0,1\}^n)^k \to \{0,1\}$ on nk bits that reads bits in any order with error $O(k^2 2^n(n + \log k)\varepsilon)$.*

Proof of Lemma 48. (1) Let $U = (U_1, \dots, U_k)$ be the uniform distribution over $(\{0,1\}^n)^k$ and $X = (X_1, \dots, X_k) = \pi G(U_\ell) \subseteq (\{0,1\}^n)^k$, where U_ℓ is uniform over $\{0,1\}^\ell$ and π is some permutation on nk bits. Let $Z_1 := \sum_{i \le k} g_i(U_i)$, and $Z_2 := \sum_{i \le k} g_i(X_i)$. Since G fools product tests with error ε, we have for every $\alpha \in [0,1]$,

$$|\mathrm{E}[e^{2\pi i\alpha Z_1}] - \mathrm{E}[e^{2\pi i\alpha Z_2}]| = |\mathrm{E}[\prod_{i \le k} e^{2\pi i\alpha g_i(U_i)}] - \mathrm{E}[\prod_{i \le k} e^{2\pi i\alpha g_i(X_i)}]| \le \varepsilon.$$

By [27, Lemma 9.3], we may assume each $g_i(j)$ and θ are integers of absoluate value $B := (2^n k)^{O(2^n k)}$, and so $-kB \leq Z_1, Z_2 \leq kB$. It follows form [27, Lemma 9.2] that

$$|\mathrm{E}[h(\pi G(U_\ell))] - \mathrm{E}[h(U)]| \leq \max_{-kB \leq t \leq kB} |\Pr(Z_1 \leq t) - \Pr(Z_2 \leq t)| \leq O(\log(kB))\varepsilon.$$

(2) Since $\sum_{i \leq k} g_i(x_i) \in \{0, \ldots, k\}$, it suffices to fool the generalized halfspaces $h(x) := \sum_{i \leq k} g_i(x_i) - \theta$ for $\theta \in \{0, \ldots, k\}$, the rest follows from (1) and a union bound. ◀

Proof of Theorem 22. Combine Lemma 48 and Theorem 19. ◀

Motivation: The sum of small-bias distributions. One motivation for this work comes from the paper [38]. That paper shows that the study of bounded independence plus noise is useful in understanding the limitations of the sum of two or more independent small-bias distributions. We refer the reader to [38] for background, but we mention briefly that while the latter distributions have been shown to fool low-degree polynomials in [15, 39, 50] they are also candidate to giving new circuit lower bounds or RL=L. The paper [38] lays two approaches to exhibit *distinguishers* for the sum of small-bias distributions, and one approach is related to this work, as discussed next.

Let D be a linear b-wise independent distribution over $\{0,1\}^m$, and for a parameter η let $E(\eta)$ be a vector of m independent bits which are set to uniform with probability η and 0 otherwise. [38] makes two observations. First, the distribution $X = D + E(\eta)$, where $+$ denotes bit-wise xor, is $\varepsilon = (1-\eta)^{b+1}$-biased. Second, by the linearity of D we have

$$X + X = D + E(\eta) + D + E(\eta) = D + E(\eta')$$

where $\eta' = 1 - (1-\eta)^2 = (2-\eta)\eta < 2\eta$. Hence, the distribution $X + X$ is of the same form as X except for a slight increase in the noise parameter. This structure is useful in exhibiting tests which are not fooled by the xor of two small-bias distributions, see [38]. However, it was left open in [38] whether it can be useful to answer a question posed more than 10 years ago by Reingold and Vadhan, and which we can state in the following form: is it true that for every c there exists a d such that the xor of two m^{-d}-biased distributions on m bits fools one-way algorithms using space $c \log m$? (An affirmative answer implies RL=L.)

The approach in [38] cannot answer this question in the negative if it turns out that whenever the bias of X is m^{-d} then $X + X$ does fools $c \log m$-space algorithms.

This paper shows that this does turn out to be the case whenever $b = \Omega(m)$, which is also the setting where our bounds are tight. Indeed, in this setting we need $\eta = \Omega(d(\log m)/m)$ to have bias m^{-d}. But then Theorem 5 gives an error bound of $2^{-\Omega(d \log m)}$. This can be made less than m^{-ck} for a constant k by choosing d large enough. And this bound is sufficient to fool one-way space $c \log m$, as remarked in §1.2 before Corollary 20.

This being the failure of an approach to show a limitation, it can be interpreted with optimism.

However, already when $b = m/\log m$ our bounds are not strong enough to show that the [38] approach fails. The bias condition gives $\eta = c(\log^2 m)/m$, and in this case our bound becomes only $2^{-\Omega(c)}$, which is not sufficient to fool space. This provides further motivation for understanding whether the bounds in Theorem 5 are tight even for $b = o(m)$, and to extend the theorem to other tests.

Bonus results. We note that for fixed order we have the following simple construction that fools product tests.

▶ **Claim 49.** *Let U_1 be the uniform distribution on n bits and $D_2, \dots, D_k$ be $k-1$ independent $\varepsilon/(k-1)$-biased distributions on n bits, then the distribution $D := (U_1, U_1 + D_2, \dots, U_1 + D_k)$ ε-fools any product test $f\colon (\{0,1\}^n)^k \to \mathbb{C}_1$.*

An ε-biased distribution can be sampled using $O(\log(n/\varepsilon))$ bits [41, 6]. Hence, D can be sampled using $n + O(k \log(nk/\varepsilon))$ bits, which is optimal when $k = O(1)$.

Proof of Claim 49. We will use the hybrid argument. Let $f := \prod_{i \le k} f_i$ be any product test and $\mu_i = \mathrm{E}[f_i]$. For $1 \le i \le k$, define the hybrid distribution $H_i := (U_1, \dots, U_i, U_1 + D_{i+1}, \dots, U_1 + D_k)$, where each U_i is independently uniformly distributed over $\{0,1\}^n$. Note that $H_1 = D$ and H_k is the uniform distribution U. The goal is to show that for $2 \le j \le k$, we have $|\mathrm{E}[H_{i-1}] - \mathrm{E}[H_i]| \le \varepsilon/(k-1)$. Then it follows that

$$|\mathrm{E}[f(D)] - \mathrm{E}[f(U)]| = |\mathrm{E}[f(H_1)] - \mathrm{E}[f(H_k)]| \le \sum_{2 \le j \le k} |\mathrm{E}[f(H_{i-1})] - \mathrm{E}[f(H_i)]| \le \varepsilon.$$

We now show that $|\mathrm{E}[H_{i-1}] - \mathrm{E}[H_i]| \le \varepsilon/(k-1)$. Note that once we have fixed the values of D_j for $j \ge i$, the corresponding f_j's has the same input as f_1. Thus we can write their products as one function. That is, for every $z_j \in D_j$ where $j > i$, we can define $g_{z_{i+1},\dots,z_k}\colon \{0,1\}^n \to \mathbb{C}_1$ by $g_{z_{i+1},\dots,z_k}(x) := f_1(x) \prod_{j>i} f_j(x + z_j)$. Then

$$\begin{aligned}
&|\mathrm{E}[f(H_{i-1})] - \mathrm{E}[f(H_i)]| \\
&= |\mathrm{E}[f_1(U_1) \cdot \prod_{j=2}^{i-1} f_j(U_j) \cdot (f_i(U_1 + D_i) - f_i(U_i)) \cdot \prod_{j>i} f_j(U_1 + D_j)]| \\
&\le (\prod_{j=2}^{i-1} \mu_j) \cdot |\mathrm{E}[(f_1(U_1) \cdot \prod_{j>i} f_j(U_1 + D_j)) \cdot (f_i(U_1 + D_i) - f_i(U_i))]| \\
&\le |\mathop{\mathrm{E}}_{z_j \sim D_j, \forall j>i} [\mathop{\mathrm{E}}_{U_1}[g_{z_{i+1},\dots,z_k}(U_1) f_i(U_1 + D_i)] - \mathrm{E}[g_{z_{i+1},\dots,z_k}] \cdot \mu_i]| \\
&\le \mathop{\mathrm{E}}_{z_j \sim D_j, \forall j>i} [|\mathop{\mathrm{E}}_{U_1}[g_{z_{i+1},\dots,z_k}(U_1) f_i(U_1 + D_i)] - \mathrm{E}[g_{z_{i+1},\dots,z_k} \cdot \mu_i]|].
\end{aligned}$$

It follows from Claim 50 below that the inner expectation is at most $\varepsilon/(k-1)$, and the rest follows by averaging over the choices of the z_j's. ◀

▶ **Claim 50.** *Let U be the uniform distribution over n bits. Let D be an ε-biased distribution over n bits. Let $f, g\colon \{0,1\}^n \to \mathbb{C}_1$ be two functions. We have $|\mathrm{E}[f(U)g(U+D)] - \mathrm{E}[f]\,\mathrm{E}[g]| \le \varepsilon$.*

Proof. We write f and g in their Fourier expansion. We have

$$\begin{aligned}
|\mathrm{E}[f(U)g(U+D)] - \mathrm{E}[f]\,\mathrm{E}[g]| &= |\mathrm{E}[(\sum_\alpha \hat{f}_\alpha \chi_\alpha(U))(\sum_\beta \hat{g}_\beta \chi_\beta(U+D))] - \hat{f}_\emptyset \hat{g}_\emptyset| \\
&= |\sum_{\alpha,\beta} \hat{f}_\alpha \hat{g}_\beta\, \mathrm{E}[\chi_{\alpha+\beta}(U)]\, \mathrm{E}[\chi_\beta(D)] - \hat{f}_\emptyset \hat{g}_\emptyset| \\
&= |\sum_{\alpha \ne \emptyset} \hat{f}_\alpha \hat{g}_\alpha\, \mathrm{E}[\chi_\alpha(D)]| \\
&\le \varepsilon \sum_{\alpha \ne \emptyset} |\hat{f}_\alpha||\hat{g}_\alpha| \\
&\le \varepsilon,
\end{aligned}$$

where the last equality is because $\mathrm{E}[\chi_\alpha(U)] = 0$ if $\alpha \ne 0$ and equals 1 otherwise, the first inequality is because D is ε-biased, and the last inequality is by Cauchy–Schwarz and the fact that f and g are bounded by 1. ◀

7 Conclusion

We have shown that distributions with bounded independence (or small-bias) perturbed with noise fool products. We ask for tight bounds on the error ε as a function of the amount of independence and the error parameter η, in any computational model. For products, an immediate question is to understand whether we can remove the factor of $1/k$ in the exponent in our main theorems. This would improve significantly our applications.

Our study also leads us to the following question. For simplicity we focus on the binary case $q = 2$.

▶ **Question.** *Let $X = (X_1, X_2, \ldots, X_k)$ be an ε-biased distribution over $(\mathbb{F}_2^n)^k$. Let U be uniform over $\mathbb{F}_2^n$. Let $f_1, f_2, \ldots, f_k$ be functions from $\{0,1\}^n \to \{0,1\}$ with expectations $\mu_1, \mu_2, \ldots, \mu_k$. Is it true that*

$$|\mathop{\mathrm{E}}_{U,X}[\prod_{i\leq k} f_i(U + X_i)] - \prod_{i\leq k} \mu_i| \leq \varepsilon'$$

for an ε' which is independent of n and that, say, goes to 0 for any fixed k and vanishing ε?

Note that the X_i may be correlated. It can be shown that the distribution $D + E$ in Theorem 1.1, when D is linear, has the above format, by writing the generator matrix in systematic form. (In fact, it is the sum of several independent samples of such distributions.)

Acknowledgments. We thank Andrej Bogdanov for useful discussions.

References

1. Miklós Ajtai. Σ_1^1-formulae on finite structures. *Annals of Pure and Applied Logic*, 24(1):1–48, 1983.
2. 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.
3. Miklos Ajtai and Avi Wigderson. Deterministic simulation of probabilistic constant-depth circuits. *Advances in Computing Research – Randomness and Computation*, 5:199–223, 1989.
4. Noga Alon, Alexandr Andoni, Tali Kaufman, Kevin Matulef, Ronitt Rubinfeld, and Ning Xie. Testing k-wise and almost k-wise independence. In *ACM Symp. on the Theory of Computing (STOC)*, pages 496–505, 2007. doi:10.1145/1250790.1250863.
5. Noga Alon, László Babai, and Alon Itai. A fast and simple randomized algorithm for the maximal independent set problem. *Journal of Algorithms*, 7:567–583, 1986.
6. 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.
7. Noga Alon, Oded Goldreich, and Yishay Mansour. Almost k-wise independence versus k-wise independence. *Inf. Process. Lett.*, 88(3):107–110, 2003.
8. 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.
9. Ziv Bar-Yossef, Omer Reingold, Ronen Shaltiel, and Luca Trevisan. Streaming through combinatorial objects. In *Seventeenth IEEE Conference on Computational Complexity*. IEEE Computer Soc., Los Alamitos, CA, 2002.
10. Louay Bazzi and Sanjoy K. Mitter. Encoding complexity versus minimum distance. *IEEE Transactions on Information Theory*, 51(6):2103–2112, 2005.

11 Louay M. J. Bazzi. Polylogarithmic independence can fool DNF formulas. *SIAM J. Comput.*, 38(6):2220–2272, 2009.

12 Anja Becker, Antoine Joux, Alexander May, and Alexander Meurer. Decoding random binary linear codes in $2^{n/20}$: How $1 + 1 = 0$ improves information set decoding. In *Int. Conf. on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, pages 520–536, 2012. doi:10.1007/978-3-642-29011-4_31.

13 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.

14 Andrej Bogdanov, Periklis A. Papakonstantinou, and Andrew Wan. Pseudorandomness for linear length branching programs and stack machines. In *Workshop on Randomization and Computation (RANDOM)*, pages 447–458, 2012.

15 Andrej Bogdanov and Emanuele Viola. Pseudorandom bits for polynomials. *SIAM J. on Computing*, 39(6):2464–2486, 2010.

16 Mark Braverman. Polylogarithmic independence fools AC^0 circuits. *J. of the ACM*, 57(5), 2010.

17 J. Lawrence Carter and Mark N. Wegman. Universal classes of hash functions. *J. of Computer and System Sciences*, 18(2):143–154, 1979.

18 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.

19 Sitan Chen, Thomas Steinke, and Salil P. Vadhan. Pseudorandomness for read-once, constant-depth circuits. *CoRR*, abs/1504.04675, 2015. URL: http://arxiv.org/abs/1504.04675.

20 Benny Chor and Oded Goldreich. On the power of two-point based sampling. *Journal of Complexity*, 5(1):96–106, 1989.

21 Anindya De. Beyond the central limit theorem: Asymptotic expansions and pseudorandomness for combinatorial sums. In *IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 883–902, 2015. doi:10.1109/FOCS.2015.59.

22 Ilias Diakonikolas, Parikshit Gopalan, Ragesh Jaiswal, Rocco A. Servedio, and Emanuele Viola. Bounded independence fools halfspaces. *SIAM J. on Computing*, 39(8):3441–3462, 2010.

23 Ilias Diakonikolas, Daniel Kane, and Jelani Nelson. Bounded independence fools degree-2 threshold functions. In *51th IEEE Symp. on Foundations of Computer Science (FOCS)*. IEEE, 2010.

24 Guy Even, Oded Goldreich, Michael Luby, Noam Nisan, and Boban Velickovic. Approximations of general independent distributions. In *ACM Symp. on the Theory of Computing (STOC)*, pages 10–16, 1992.

25 Guy Even, Oded Goldreich, Michael Luby, Noam Nisan, and Boban Velickovic. Efficient approximation of product distributions. *Random Struct. Algorithms*, 13(1):1–16, 1998.

26 Merrick L. Furst, James B. Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17(1):13–27, 1984.

27 Parikshit Gopalan, Daniek 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.

28 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.

29 Parikshit Gopalan, Raghu Meka, Omer Reingold, and David Zuckerman. Pseudorandom generators for combinatorial shapes. *SIAM J. on Computing*, 42(3):1051–1076, 2013. doi:10.1137/110854990.

30 Parikshit Gopalan, Ryan O'Donnell, Yi Wu, and David Zuckerman. Fooling functions of halfspaces under product distributions. In *25th IEEE Conf. on Computational Complexity (CCC)*, pages 223–234. IEEE, 2010.

31 Parikshit Gopalan and Amir Yehudayoff. Inequalities and tail bounds for elementary symmetric polynomials. *Electronic Colloquium on Computational Complexity (ECCC)*, 21:19, 2014. URL: https://eccc.weizmann.ac.il/report/2014/019/.

32 Andre Gronemeier. A note on the decoding complexity of error-correcting codes. *Information Processing Letters*, 100(3):116–119, 2006. doi:10.1016/j.ipl.2006.06.006.

33 Johan Håstad. *Computational limitations of small-depth circuits*. MIT Press, 1987.

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

35 Russell Impagliazzo, William Matthews, and Ramamohan Paturi. A satisfiability algorithm for AC^0. In *ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 961–972, 2012.

36 Russell Impagliazzo, Raghu Meka, and David Zuckerman. Pseudorandomness from shrinkage. In *IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 111–119, 2012.

37 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.

38 Chin Ho Lee and Emanuele Viola. Some limitations of the sum of small-bias distributions. To appear in Theory of Computing, pre-print available at https://eccc.weizmann.ac.il/report/2015/005/.

39 Shachar Lovett. Unconditional pseudorandom generators for low degree polynomials. *Theory of Computing*, 5(1):69–82, 2009. arXiv:toc:v005/a003, doi:10.4086/toc.2009.v005a003.

40 Chi-Jen Lu. Improved pseudorandom generators for combinatorial rectangles. *Combinatorica*, 22(3):417–433, 2002.

41 Joseph Naor and Moni Naor. Small-bias probability spaces: efficient constructions and applications. *SIAM J. on Computing*, 22(4):838–856, 1993.

42 Noam Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992.

43 Noam Nisan and David Zuckerman. Randomness is linear in space. *J. of Computer and System Sciences*, 52(1):43–52, February 1996.

44 Alexander A. Razborov. A simple proof of Bazzi's theorem. *ACM Transactions on Computation Theory (TOCT)*, 1(1), 2009.

45 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.

46 Amir Shpilka. Constructions of low-degree and error-correcting epsilon-biased generators. *Computational Complexity*, 18(4):495–525, 2009.

47 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.

48 B. A. Subbotovskaya. Realizations of linear functions by formulas using $+$, $*$, $-$. *Soviet Mathematics-Doklady*, 2:110–112, 1961.

49 Avishay Tal. Tight bounds on The Fourier Spectrum of AC^0, 2014. Electronic Colloquium on Computational Complexity, Technical Report TR14-174. URL: https://eccc.weizmann.ac.il/report/2014/174/.

50 Emanuele Viola. The sum of d small-bias generators fools polynomials of degree d. *Computational Complexity*, 18(2):209–217, 2009.

51 Emanuele Viola. The complexity of distributions. *SIAM J. on Computing*, 41(1):191–218, 2012.

52 Emanuele Viola. Randomness buys depth for approximate counting. *Computational Complexity*, 23(3):479–508, 2014.

53 Thomas Watson. Pseudorandom generators for combinatorial checkerboards. *Computational Complexity*, 22(4):727–769, 2013. doi:10.1007/s00037-012-0036-6.

54 Andrew Yao. Separating the polynomial-time hierarchy by oracles. In *26th IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 1–10, 1985.

A A lower bound on b and η

For completeness we now prove by standard arguments a lower bound on b and η such that a result as in Theorem 5 may apply. We obviously require $b \geq n$ if $\eta = 0$, whereas for general η one requires a more elaborate argument. Let $k = 1$ and let M be a uniformly chosen $n \times t$ matrix over $\mathbb{F}_q$. The probability that the corresponding code has minimum distance $\leq d$ is at most $q^t V_q(d)/q^n$. Hence a code C' exists with minimum distance $> d$ for $n - t = \lceil \log_q V_q(d) \rceil$. By Fact 11 the uniform distribution D over the dual C of C' is d-uniform. This distribution can be generated by an $n \times (n-t)$ matrix. Hence the support size of this distribution is $q^{n-t} \leq O(V_q(d))$.

Moreover, by Lemma 34 we can sample with $O(\eta \log(q/\eta)n)$ bits a distribution that is $2^{-\Omega(\eta n)}$-close to the noise vector E.

Hence $D + E$ is $2^{-\Omega(\eta n)}$-close to a distribution supported on a set S whose size is $O(V_q(d))2^{O(\eta \log(q/\eta)n)} \leq 2^{d \log O(enq/d) + O(\eta \log(q/\eta)n)}$. The function f_1 is taken to be the characteristic function of S. By the lemma the function outputs 1 on $D + E$ with probability $1 - 2^{-\Omega(\eta n)}$.

On the other hand, the function outputs 1 on a uniform input with probability $1 - |S|/q^n$. In particular, for any $d = (1 - \varepsilon)n$ and sufficiently large q this shows that f_1 has a constant distinguishing advantage for all η less ε', where ε' depends only on ε.

Tight Bounds on the Fourier Spectrum of AC^0*

Avishay Tal

Institute for Advanced Study, Princeton, NJ, USA
avishay.tal@gmail.com

Abstract

We show that $\mathbf{AC^0}$ circuits on n variables with depth d and size m have at most $2^{-\Omega(k/\log^{d-1} m)}$ of their Fourier mass at level k or above. Our proof builds on a previous result by Håstad (SICOMP, 2014) who proved this bound for the special case $k = n$. Our result improves the seminal result of Linial, Mansour and Nisan (JACM, 1993) and is tight up to the constants hidden in the Ω notation.

As an application, we improve Braverman's celebrated result (JACM, 2010). Braverman showed that any $r(m, d, \varepsilon)$-wise independent distribution ε-fools $\mathbf{AC^0}$ circuits of size m and depth d, for

$$r(m, d, \varepsilon) = O(\log(m/\varepsilon))^{2d^2+7d+3}.$$

Our improved bounds on the Fourier tails of $\mathbf{AC^0}$ circuits allows us to improve this estimate to

$$r(m, d, \varepsilon) = O(\log(m/\varepsilon))^{3d+3}.$$

In contrast, an example by Mansour (appearing in Luby and Velickovic's paper – Algorithmica, 1996) shows that there is a $\log^{d-1}(m)\cdot\log(1/\varepsilon)$-wise independent distribution that does not ε-fool $\mathbf{AC^0}$ circuits of size m and depth d. Hence, our result is tight up to the factor 3 in the exponent.

1998 ACM Subject Classification F.1.3 [Computation by Abstract Devices] Complexity Measures and Classes

Keywords and phrases bounded depth circuits, Fourier analysis, k-wise independence, Boolean circuits, switching lemma

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.15

1 Introduction

In this paper we discuss Boolean circuits in which every gate computes an unbounded fan-in OR or AND function of its inputs, and every leaf is marked with a literal from $x_1, \ldots, x_n, \neg x_1, \ldots, \neg x_n$. The number of gates in the circuit is called the circuit size and is denoted by m. The longest path in the circuit is called the circuit depth and is denoted by d. $\mathbf{AC^0}$ is the class of functions that can be realized by Boolean circuits of constant depth and polynomial size. (We also call Boolean circuits of polynomial size and constant depth $\mathbf{AC^0}$ circuits).

The study of bounded depth circuits flourished in the 1980s, culminating in the tight $\exp(\Omega(n^{1/(d-1)}))$ size lower bound for Boolean circuits of depth d computing the parity function [2, 11, 36, 14].[1] The main idea behind this lower bound was the following – Boolean

* Work done while the author was a student at Weizmann Institute of Science, Rehovot, Israel. Research supported by an Adams Fellowship of the Israel Academy of Sciences and Humanities, by an ISF grant and by the I-CORE Program of the Planning and Budgeting Committee.

[1] Lower bounds for the **DNF**-size of the parity function were known long before [24].

circuits with size m and depth d become constant with high probability under random restrictions keeping each variable alive with probability $p = 1/O(\log m)^{d-1}$. In contrast, the parity function does not become a constant with probability at least 0.5 as long as $pn \geq 1$. Since the restricted circuit should compute the restricted function, we reach a contradiction if $m = \exp(o(n^{1/(d-1)}))$. The main idea is carried through a sequence of $d-1$ steps, where in each step the circuit depth is decreased by one with high probability, by applying Håstad's switching lemma [14].

In their seminal paper, Linial, Mansour, and Nisan [22] showed that $\mathbf{AC^0}$ circuits can be learned in quasipolynomial time, $n^{O(\log^d n)}$, using random samples, under the uniform distribution. They combined Håstad's switching lemma with Fourier analysis, to show that $\mathbf{AC^0}$ circuits may be well approximated (in L_2 norm) by low degree polynomials, namely polynomials of degree $O(\log^d n)$. Boppana [6] improved their bound on the degree to $O(\log^{d-1} n)$, which is optimal for constant error. The existence of an approximating low degree polynomial implies a learning algorithm for $\mathbf{AC^0}$ circuits, using random examples. For polynomial size **DNFs** (depth 2 circuits), Mansour [25] showed that only $n^{O(\log\log n)}$ out of the $\binom{n}{\leq O(\log n)}$ monomials are needed to approximate the **DNF**, and achieved a $n^{O(\log\log n)}$ time learning algorithm for **DNFs**, using membership queries, via the Goldreich-Levin [12], Kushilevitz-Mansour [21] method.

The main technical result in [22] was a bound on the Fourier tails of Boolean circuits. Namely, for any circuit f of size m and depth d,

$$\sum_{S\subseteq[n]:|S|\geq k} \hat{f}(S)^2 \leq m \cdot 2^{-\Omega(k^{1/d})} ,$$

where the LHS is called the Fourier tail of f at level k. This was later improved by Håstad [15] to

$$\sum_{S\subseteq[n]:|S|\geq k} \hat{f}(S)^2 \leq \max\{2^{-\Omega((k/\log m)^{1/(d-1)})}, 2^{-\Omega(k/\log^{d-1}(m))}\} ,$$

which is tight for $k \leq O(\log^d(m))$, however not for larger values of k. Recently, Håstad [16] and Impagliazzo, Matthews, and Paturi [18] showed that any Boolean circuit f agrees with parity on at most a $1/2 + 2^{-n/O(\log m)^{d-1}}$ fraction of the inputs. In other words, they showed that $|\hat{f}([n])| \leq 2^{-n/O(\log m)^{d-1}}$.

1.1 Our Results

Based on the main lemma of [16], we extend the results of [16, 18] for all $k \in [0, n]$ and show the following.

▶ **Theorem 1** (Main Theorem). *Let f be an Boolean circuit with depth d and size m. Then,*

$$\sum_{S:|S|\geq k} \hat{f}(S)^2 \leq 2 \cdot 2^{-k/O(\log m)^{d-1}} .$$

A few things to note first. Increasing k from 0 to n, the first time that Theorem 1 is meaningful is at $k = \Theta(\log^{d-1}(m))$, which is only marginally better than in [22] and exactly the same as in [6, 15]. Nonetheless, for larger values, our bound decreases much faster, and in particular for $m = \text{poly}(n)$ we get a $2^{-n/\text{poly}\log(n)}$ tail at level $k = \Omega(n)$ as opposed to a $2^{-\Omega((n/\log n)^{1/(d-1)})}$ tail by [15]. In addition, while [16] and [18] give bounds on an individual coefficient, $|\hat{f}(S)|$, we give bounds on the sum of $\exp(\Omega(n))$ many squares of coefficients (e.g., for $k = n/2$).

We point out that the results of [16], [18], and ours are quite surprising, considering the fact that most proofs for Boolean circuits follow by induction on the depth d; performing $d-1$ consecutive steps of Håstad's switching lemma. Our main theorem is equivalent to saying that degree $O(\log^{d-1}(m)\cdot\log(1/\varepsilon))$ polynomials ε-approximates Boolean circuits of size m and depth d, as opposed to degree $O(\log^{d}(m/\varepsilon))$ polynomials by [22]. It seems at first glance that one *must* pay a factor of $\log(m/\varepsilon)$ for each step in the induction to ensure error at most ε, thus resulting in degree at least $\log^{d-1}(m/\varepsilon)$. However, Håstad and Impagliazzo et al. managed to avoid that. Håstad performs random restrictions keeping each variable alive with probability $p=1/O(\log m)$ that does not depend on ε. This only guarantee that the switching succeeds with probability $1-1/\mathrm{poly}(m)$, as opposed to probability of $1-\varepsilon/m$ in the original proof of [22]. However, in the cases where the switching "fails", Håstad fixes D additional variables using a decision tree of depth D. Under these additional fixings, the probability that the switching fails reduces to $m\cdot 2^{-D}$. We show that the parameters p and D translate into a **multiplicative** term of $1/p$ and an **additive** term of D in the degree, correspondingly. Choosing D to be roughly $\log(m/\varepsilon)$ and applying induction gives the desired dependency on m and ε.

Theorem 1 shows that the Fourier tail above level k decreases exponentially fast in k. In Section 5, we show that such behavior is related to three other properties of concentration. We establish many connections between these four properties, and show that three of them are essentially equivalent. We think that these connections are of independent interest.[2] As a result of these connections we establish the following theorem.

▶ **Theorem 2.** *Let f be an Boolean circuit with depth d and size m. Then,*

1. *For all k,p, if ρ is a p-random restriction, then $\boldsymbol{Pr}_{\rho}[\deg(f|_{\rho})\geq k]\leq O(p\cdot\log^{d-1}(m))^{k}$.*
2. *For all k,*

$$\sum_{S:|S|=k}|\hat{f}(S)|\leq O(\log^{d-1}(m))^{k}\ . \tag{1}$$

3. *f is ε-concentrated on at most $2^{O(\log\log(m)\cdot\log^{d-1}(m)\cdot\log(1/\varepsilon))}$ Fourier coefficients.*

In Section 6, we show that Equation (1) gives new proofs for the following known results:

- Correlation bounds for the Majority function. If f is a size m depth d circuit, then $\mathbf{Pr}[f(x)=\mathrm{MAJ}(x)]\leq\frac{1}{2}+\frac{O(\log^{d-1}(m))}{\sqrt{n}}$. Our result holds for $\log^{d-1}(m)=O((n/\log n)^{1/3})$, which is an artifact of the proof. This result was originally proved by Smolensky [32] (see also [10]) and by O'Donnell and Wimmer [28], for the entire range of parameters.
- Boolean circuits cannot distinguish between fair coins and coins with bias at most $\frac{1}{O(\log^{d-1}(m))}$. This result was previously proved by Cohen, Ganor and Raz [8], improving the results of Aaronson [1], and Shaltiel and Viola [31].

1.2 Applications to Pseudorandomness and Learning

Since the result of [22] had many applications, our main theorem improves some of them as well.

[2] In fact, some of these connections have been already used in the context of de Morgan formulae [33].

***k*-wise independence fools bounded-depth circuits.** The most significant improvement is to the work of Braverman [7] who proved a longstanding conjecture, showing that poly-logarithmic independent distributions fool $\mathbf{AC^0}$ circuits. To be more precise, Braverman showed that any k-wise independent distribution, where $k = O(\log(m/\varepsilon))^{2d^2+7d+3}$, ε-fools circuits of size m and depth d. In addition, it was long known [23] that k must be larger than $\Omega(\log^{d-1}(m) \cdot \log(1/\varepsilon))$; otherwise, there is a k-wise independent distribution that is ε-distinguishable from the uniform distribution by a depth d, size m circuit. Our theorem improves Braverman's bounds to $k = O\left(\log(m/\varepsilon)\right)^{3d+3}$, answering an open question posed by Braverman on the affirmative. In particular, our result is non-trivial for polynomial size circuits of depth $d \leq 0.3\log(n)/\log\log(n)$. Since $\mathbf{NC^1}$ circuits can be computed by Boolean circuits of depth $O(\log(n)/\log\log(n))$ and polynomial size, constructing a non trivial PRG for all $d = O(\log(n)/\log\log(n))$ is a major open challenge. While the dependence of k on m and d is close to optimal, we conjecture that the dependence on ε could be much better.[3]

▶ **Conjecture 3.** *Any k-wise independence ε-fools circuits of size m and depth d, for*

$$k = (\log m)^{O(d)} \cdot \log(1/\varepsilon) \ .$$

***k*-wise independence fools DNFs.** We improve in Section 4.2 the earlier result of Bazzi [4], who showed that $O(\log^2(m/\varepsilon))$-wise independence ε-fools **DNFs** of size m. We improve the dependence on ε and get that $O(\log(m) \cdot \log(m/\varepsilon))$-wise independence suffices. Note that by [23] this is optimal for $\varepsilon \leq 1/m^{\Omega(1)}$. The range $\varepsilon \geq 1/m^{o(1)}$ is still not tightly understood.

PRGs for $\mathbf{AC^0}$ and DNFs. We improve the results of De et al. [9] (see Appendix C) and of Trevisan and Xue [35] (see Appendix D) that give the best known PRGs for **DNFs** and $\mathbf{AC^0}$ circuits respectively. In the PRG of De et al., we improve the dependency of the seed-length in ε, as seen in Figure 1. Since Trevisan and Xue used De et al.'s generator as a black-box in their construction, we also improve the seed length of their PRG for $\mathbf{AC^0}$ circuits. We observe two more improvements in the Trevisan-Xue generator to reduce the seed-length to $\tilde{O}(\log^{d+1}(m/\varepsilon) \cdot \log(n))$. This seed-length comes closer to the barrier $O(\log^d(m/\varepsilon))$ noted by [35].

Sparse polynomial approximations of Boolean circuits. Theorem 2 shows that any Boolean circuit f of size m and depth d can be ε-approximated in L_2 by a polynomial $p(x)$ of sparsity $(\log m)^{O(\log^{d-1}(m)\cdot\log(1/\varepsilon))}$, improving the results of [22] and [25]. As the inner product on $k = \log^{d-1} m$ variables can be realized by a size poly(m) depth d circuit, and requires at least $\Omega(2^k)$ coefficients in order to $\Omega(1)$ approximate in L_2, one cannot achieve sparsity $2^{o(\log^{d-1} m)}$.

A table summarizing all of the improvements mentioned above is presented in Figure 1.

1.3 Organization

In Section 2, we lay out some preliminary definitions and results that will be used in the rest of the paper. In Section 3, we prove our main theorem, i.e. Theorem 1. In Section 4, we improve Braverman's and Bazzi's results in the field of pseudorandomness. In Section 5, we prove Theorem 2, by relating different notions of Fourier concentration. Then, in section 6, we use Theorem 2 to deduce simpler proofs for two known results: the inapproximability of

[3] We have learned that subsequent to this work, Harsha and Srinivasan [13] proved this conjecture.

Task	Reference	Bound
k-wise ind. fooling **DNFs**	[4]	$k = O(\log^2(m/\varepsilon))$
	This Work	$k = O(\log(m/\varepsilon) \cdot \log(m))$
	Lower Bound	$k \geq \log(m) \cdot \log(1/\varepsilon)$
k-wise ind. fooling $\mathbf{AC^0}$	[7]	$k = O((\log(m/\varepsilon))^{d^2+3d} \cdot (\log m)^{d^2+4d+3})$
	This Work	$k = O\left((\log(m/\varepsilon))^{d} \cdot (\log m)^{2d+3}\right)$
	Lower Bound	$k \geq \log^{d-1}(m) \cdot \log(1/\varepsilon)$
sparse polynomial approximating **DNFs** in L_2	[25]	sparsity $= (m/\varepsilon)^{O(\log\log(m/\varepsilon)\cdot\log(1/\varepsilon))}$
	This Work	sparsity $= m^{O(\log\log(m)\cdot\log(1/\varepsilon))}$
sparse polynomial approximating $\mathbf{AC^0}$ in L_2	[22]	sparsity $= 2^{O(\log(n)\cdot\log^{d}(m/\varepsilon))}$
	[15]	sparsity $= 2^{O(\log(n)\cdot\log^{d-2}(m/\varepsilon)\cdot\log(m)\cdot\log(1/\varepsilon))}$
	This Work	sparsity $= 2^{O(\log\log(m)\cdot\log^{d-1}(m)\cdot\log(1/\varepsilon))}$
	Lower Bound	sparsity $\geq 2^{\Omega(\log^{d-1}(m))}$
PRGs for **DNFs**	[9]	seed $= O(\log n + \log^2(m/\varepsilon) \cdot \log\log(m/\varepsilon))$
	This Work	seed $= O(\log n + \log(m/\varepsilon) \cdot \log(m) \cdot \log\log m)$
PRGs for $\mathbf{AC^0}$	[35]	seed $= \tilde{O}(\log^{d+4}(m/\varepsilon))$
	This Work	seed $= \tilde{O}(\log^{d+1}(m/\varepsilon) \cdot \log n)$

Figure 1 Summary of Applications.

the Majority function by bounded-depth circuits, and the indistinguishability of biased-coins from uniform coins by bounded-depth circuits. In Section 7, we give a self-contained new proof of the main lemma in the work of Håstad [16], that plays a crucial role in the proof of Theorem 1. This serves two purposes. First, it makes the main result in our paper self-contained. Second, in our opinion, it gives a simpler proof of Håstad's main lemma ([16]).

In the appendices, we revisit the works of Braverman [7] (Appendix B), De et al. [9] (Appendix C), and Trevisan and Xue [35] (Appendix D) in the field of pseudorandomness. We show how our main results (Theorem 1 and Theorem 2) improve these results. Furthermore, we reduce the seed-length of the PRG of [35] even further using several other observations.

2 Preliminaries

We denote by $[n] = \{1, \ldots, n\}$. We denote by log and ln the logarithms in bases 2 and e, respectively. For $f : \{-1,1\} \to \mathbb{R}$ we denote by $\|f\|_p = \left(\mathbf{E}_{x\in\{-1,1\}^n}[|f(x)|^p]\right)^{1/p}$.

2.1 Restrictions

▶ **Definition 4** (Restriction)**.** Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function. A restriction ρ is a vector of length n of elements from $\{0,1,*\}$. We denote by $f|_\rho : \{0,1\}^n \to \{0,1\}$ the function f restricted according to ρ, defined by

$$f|_\rho(x) = f(y), \quad \text{where} \quad y_i = \begin{cases} x_i, & \rho_i = * \\ \rho_i, & \text{otherwise} \end{cases}.$$

We say that the variable x_i is fixed if $\rho_i \in \{0,1\}$, and that x_i is unassigned (or alive) if $\rho_i = *$.

Note that the function $f|_\rho$ is defined as a function with n variables, although it depends only on the non-fixed variables. When fixing only one bit to a constant, we may denote the restricted function by $f|_{x_i=b}$.

▶ **Definition 5** (p-Random Restriction)**.** A p-random restriction is a restriction as in Definition 4 that is sampled in the following way. For every $i \in [n]$, independently, with probability p set $\rho_i = *$ and with probability $\frac{1-p}{2}$ set ρ_i to be -1 and 1, respectively. We denote this distribution of restrictions by $\mathcal{R}_p$.

2.2 Fourier Analysis of Boolean Functions

Any function $f : \{-1,1\}^n \to \mathbb{R}$ has a unique Fourier representation:

$$f(x) = \sum_{S \subseteq [n]} \hat{f}(S) \cdot \prod_{i \in S} x_i \ ,$$

where the coefficients $\hat{f}(S) \in \mathbb{R}$ are given by $\hat{f}(S) = \mathbf{E}_x[f(x) \cdot \prod_{i \in S} x_i]$. Parseval's identity states that $\sum_S \hat{f}(S)^2 = \mathbf{E}_x[f(x)^2] = \|f\|_2^2$, and in the case that f is Boolean (i.e., $f : \{-1,1\}^n \to \{-1,1\}$), all are equal to 1. The Fourier representation is the unique multilinear polynomial which agrees with f on $\{-1,1\}^n$. We denoted by $\deg(f)$ the degree of this polynomial, which also equals $\max\{|S| : \hat{f}(S) \neq 0\}$. We denote by

$$\mathbf{W}^k[f] \triangleq \sum_{S \subseteq [n], |S|=k} \hat{f}(S)^2$$

the Fourier weight at level k of f. Similarly, we denote $\mathbf{W}^{\geq k}[f] \triangleq \sum_{S \subseteq [n], |S| \geq k} \hat{f}(S)^2$. The truncated Fourier expansion of degree k of f is simply $f^{\leq k}(x) = \sum_{|S| \leq k} \hat{f}(S) \prod_{i \in S} x_i$. By Parseval, $\|f - f^{\leq k}\|_2^2 = \mathbf{W}^{\geq k+1}[f]$. The following fact relates the Fourier coefficients of f and $f|_\rho$, where ρ is a p-random restriction.[4]

▶ **Fact 6** (Proposition 4.17, [27])**.** *Let $f : \{-1,1\}^n \to \mathbb{R}$, $S \subseteq [n]$, and $p > 0$. Then,*

$$\mathop{\boldsymbol{E}}_{\rho \sim \mathcal{R}_p} \left[\widehat{f|_\rho}(S) \right] = \hat{f}(S) p^{|S|}$$

and

$$\mathop{\boldsymbol{E}}_{\rho \sim \mathcal{R}_p} \left[\widehat{f|_\rho}(S)^2 \right] = \sum_{U \subseteq [n]} \hat{f}(U)^2 \cdot \mathop{\boldsymbol{Pr}}_{\rho \sim \mathcal{R}_p} [\{i \in U : \rho(i) = *\} = S] \ .$$

Summing the last equation over all sets S of size d gives the following corollary.

▶ **Fact 7.** *Denote by $\mathrm{Bin}(k,p)$ a binomial random variable with parameters k and p. Then,*

$$\mathop{\boldsymbol{E}}_{\rho \sim \mathcal{R}_p} \left[\mathbf{W}^d[f|_\rho] \right] = \sum_{k=d}^{n} \mathbf{W}^k[f] \cdot \boldsymbol{Pr}[\mathrm{Bin}(k,p) = d] \ .$$

▶ **Definition 8** (Fourier Sparsity, Spectral Norm)**.** We define the sparsity of $f : \{-1,1\}^n \to \mathbb{R}$ as $\mathrm{sparsity}(f) \triangleq |\{S : \hat{f}(S) \neq 0\}|$; the spectral norm of f as $L_1(f) \triangleq \sum_S |\hat{f}(S)|$; and the spectral norm of the k-th level of f as $L_{1,k}(f) \triangleq \sum_{S:|S|=k} |\hat{f}(S)|$.

[4] Note that $\widehat{f|_\rho}(S) = 0$ if ρ fixes one of the variables in S.

We state the following known fact regarding the Fourier sparsity, spectral norm and granularity of low degree Boolean functions.

▶ **Fact 9** (Ex. 1.11, [27]). *Let* $f : \{-1,1\}^n \to \{-1,1\}$ *with* $\deg(f) = d$. *Then:*
1. $\forall S : |\hat{f}(S)| = k_S \cdot 2^{-d}$ *where* $k_S \in \mathbb{Z}$.
2. $\text{sparsity}(f) \leq 2^{2d}$.
3. $L_1(f) \leq 2^d$.

3 Exponentially Small Fourier Tails for Bounded Depth Circuits

We generalize the proof of Håstad ([16]), who showed that the correlation between the parity function and any Boolean circuit of depth d and size m is at most $2^{-\Omega(n/\log^{d-1}(m))}$. This bound is tight up to the constants in the exponent, as shown by an example in [16], and improves upon previous bounds from [22, 15].

We will use two simple lemmata which explain the behavior of Fourier tails with respect to random restrictions, and arbitrary restrictions.

▶ **Lemma 10** ([22]). *For any* $f : \{-1,1\}^n \to \mathbb{R}$, $k \in \mathbb{N} \cup \{0\}$ *and* $p \in [0,1]$,

$$\mathbf{W}^{\geq k}[f] \leq 2 \cdot \mathop{\boldsymbol{E}}_{\rho \sim \mathcal{R}_p} \left[\mathbf{W}^{\geq \lfloor kp \rfloor}[f|_\rho] \right] .$$

Proof. Let $k \in \mathbb{N} \cup \{0\}$ and $p \in [0,1]$. We have

$$\begin{aligned}
\mathop{\mathbf{E}}_{\rho \sim \mathcal{R}_p} \left[\mathbf{W}^{\geq \lfloor kp \rfloor}[f|_\rho] \right] &= \sum_{\ell \geq \lfloor kp \rfloor} \mathbf{W}^{\ell}[f] \cdot \mathbf{Pr}[\text{Bin}(\ell, p) \geq \lfloor kp \rfloor] && \text{(Fact 7)} \\
&\geq \sum_{\ell \geq k} \mathbf{W}^{\ell}[f] \cdot \mathbf{Pr}[\text{Bin}(\ell, p) \geq \lfloor kp \rfloor] \\
&\geq \sum_{\ell \geq k} \mathbf{W}^{\ell}[f] \cdot 1/2 && (\text{median}(\text{Bin}(\ell, p)) \geq \lfloor \ell p \rfloor \geq \lfloor kp \rfloor, [19]) \\
&= 1/2 \cdot \mathbf{W}^{\geq k}[f]. && \blacktriangleleft
\end{aligned}$$

The second lemma, taken from [17], states that if, for some bit, we have Fourier tail bounds for both restrictions fixing that bit to either $+1$ or -1, then we have Fourier tail bounds for the unrestricted function.

▶ **Lemma 11** ([17]). *Let* $f : \{-1,1\}^n \to \mathbb{R}$ *and* $i \in [n]$. *Then,*

$$\mathbf{W}^{\geq k}[f] \leq \frac{1}{2} \cdot \mathbf{W}^{\geq k-1}[f|_{x_i=-1}] + \frac{1}{2} \cdot \mathbf{W}^{\geq k-1}[f|_{x_i=1}].$$

In order to generalize the last lemma, we introduce the following definition, which is very similar to the definition of a decision tree, except we are not making any decision!

▶ **Definition 12** (Restriction Tree). A restriction tree is a rooted directed binary tree such that each internal node is labeled by a variable from $x_1, \ldots, x_n$ and has two outgoing edges: one marked with 1 and one marked with -1. The leaves of the tree are not labeled. Each leaf in the tree, ℓ, corresponds to a restriction τ_ℓ on the variables $x_1, \ldots, x_n$ in the most natural way: we fix the variables along the path from the root to the leaf ℓ according to the values on the path edges.

Using induction, Lemma 11 implies (informally) that if, for some restriction tree, we have Fourier tail bounds for restrictions corresponding to all root-leaf paths in the tree, then we have Fourier tail bounds for the unrestricted function as well. The exact statement follows.

▶ **Lemma 13.** *Let $f : \{-1,1\}^n \to \mathbb{R}$ be a function, and let T be a restriction tree of depth $\leq D$ such that for any leaf ℓ, under the corresponding restriction $\mathbf{W}^{\geq k}[f|_{\tau_\ell}] \leq \varepsilon$. Then, $\mathbf{W}^{\geq k+D}[f] \leq \varepsilon$.*

Proof. Apply induction on the depth of the restriction tree. For depth 0 this obviously holds. For depth D, consider both subtrees that are rooted by the children of the original root. If the root queries x_i, these are restriction trees for $\{x : x_i = 1\}$ and $\{x : x_i = -1\}$, and we may apply the induction hypothesis on each subtree to get $\mathbf{W}^{\geq k+(D-1)}[f|_{x_i=1}] \leq \varepsilon$ and $\mathbf{W}^{\geq k+(D-1)}[f|_{x_i=-1}] \leq \varepsilon$. Finally, applying Lemma 11 gives $\mathbf{W}^{\geq k+D}[f] \leq \frac{\varepsilon}{2} + \frac{\varepsilon}{2} = \varepsilon$. ◀

Our proof relies on the main lemma in Håstad's work [16]. We begin with a definition from [16] and the statement of his main lemma.

▶ **Definition 14** (Common Partial Decision Tree)**.** A set of functions $(g_i)_{i=1}^m$ has a common s-partial decision tree of depth D, if there is a restriction tree of depth D such that at each leaf ℓ of this restriction tree, each function g_i, restricted by τ_ℓ, is computable by an ordinary decision tree of depth s.

▶ **Lemma 15** ([16], Lemma 3.8)**.** *Let $(f_i)_{i=1}^m$ be a collection of depth-2 circuits, each of bottom fan-in t. Let ρ be a random restriction from $\mathcal{R}_p$. Then the probability that $(f_i|_\rho)_{i=1}^m$ is not computable by a common $\log(2m)$-partial decision tree of depth D is at most $m \cdot (24pt)^D$.*

In Appendix 7 we give a new proof for Lemma 15 (with constant 49 instead of 24) following the proof approach of [29], [5] and [34] for the original switching lemma.

We are ready to prove the Fourier tail bounds for Boolean circuits. We define the effective size of a Boolean circuit as the number of gates in the circuit at distance 2 or more from the inputs.

▶ **Theorem 16.** *Let f be a Boolean circuit of depth d, effective size m, and bottom fan-in t. Then, $\mathbf{W}^{\geq k}[f] \leq 8^{d-1} \cdot 2^{-k/\left(20t(96\log(2m))^{d-2}\right)}$.*

Proof. We prove by induction on d. The base case $d = 2$ was proved by Mansour [25], who showed that **DNFs** with bottom fan-in t have

$$\mathbf{W}^{\geq k}[f] \leq 4 \cdot 2^{-k/20t} .$$

For the induction step, we apply a p-random restriction with $p = 1/48t$. Consider the gates at distance 2 from the inputs: $f_1, \ldots, f_{m'}$, for $m' \leq m$. These gates compute functions given by depth-2 circuits with bottom fan-in $\leq t$. Setting $D = \lfloor kp/2 \rfloor$ and using Lemma 15 gives that with probability at least $1 - m \cdot 2^{-D} \geq 1 - 2^{\log(m)-D}$ over the random restrictions, $(f_i|_\rho)_{i=1}^{m'}$ can be computed by a common $\log(2m)$-partial decision tree of depth D. In this case, we say that the restriction ρ is *good*. Using Lemma 10 we have $\mathbf{W}^{\geq k}[f] \leq 2 \cdot \mathbf{E}_\rho[\mathbf{W}^{\geq \lfloor kp \rfloor}[f|_\rho]]$. Since $\mathbf{W}^{\geq \lfloor kp \rfloor}[f|_\rho]$ is a random variable bounded in $[0,1]$ we have

$$\begin{aligned}
\mathbf{W}^{\geq k}[f] &\leq 2 \cdot \mathop{\mathbf{E}}_{\rho \sim \mathcal{R}_p}[\mathbf{W}^{\geq \lfloor kp \rfloor}[f|_\rho]] \\
&= 2 \cdot \mathop{\mathbf{E}}_{\rho \sim \mathcal{R}_p}\left[\mathbf{W}^{\geq \lfloor kp \rfloor}[f|_\rho] \mid \rho \text{ is good}\right] \cdot \mathop{\mathbf{Pr}}_{\rho \sim \mathcal{R}_p}[\rho \text{ is good}] \\
&\quad + 2 \cdot \mathop{\mathbf{E}}_{\rho \sim \mathcal{R}_p}\left[\mathbf{W}^{\geq \lfloor kp \rfloor}[f|_\rho] \mid \rho \text{ is bad}\right] \cdot \mathop{\mathbf{Pr}}_{\rho \sim \mathcal{R}_p}[\rho \text{ is bad}] \\
&\leq 2 \cdot \mathop{\mathbf{E}}_{\rho \sim \mathcal{R}_p}\left[\mathbf{W}^{\geq \lfloor kp \rfloor}[f|_\rho] \mid \rho \text{ is good}\right] + 2 \cdot \mathop{\mathbf{Pr}}_{\rho \sim \mathcal{R}_p}[\rho \text{ is bad}] ,
\end{aligned}$$

where $\mathbf{Pr}_\rho[\rho \text{ is bad}] \leq 2^{\log(m)-\lfloor k/96t \rfloor} \leq 2^{\log(2m)-k/96t}$. Using the following simple claim, we get $\mathbf{Pr}_\rho[\rho \text{ is bad}] \leq 2 \cdot 2^{-k/(96t\log(2m))}$.

▶ **Claim 17.** *If $0 \leq X \leq 1$ and $X \leq 2^{a-b}$, where $a \geq 1$, then $X \leq 2^{1-b/a}$.*

Proof. Since $0 \leq X \leq 1$ and $a \geq 1$, we have $X \leq X^{1/a}$, and $X^{1/a}$ is at most $2^{1-b/a}$. ◀

We are left to analyze $\mathbf{E}[\mathbf{W}^{\geq \lfloor kp \rfloor}[f|_\rho] \mid \rho \text{ is good}]$. Fixing ρ to be some specific good restriction, we will bound $\mathbf{W}^{\geq \lfloor kp \rfloor}[f|_\rho]$ for this specific ρ. By the definition of good restrictions, we have a common $\log(2m)$-partial decision tree of depth $D = \lfloor kp/2 \rfloor$ computing $(f_i|_\rho)_{i=1}^{m'}$. For each leaf ℓ of the common partial decision tree, let τ_ℓ be the restriction defined by the path leading to this leaf. We have that $f_i|_\rho|_{\tau_\ell}$ for $i = 1, \ldots, m'$ can be expressed as a decision tree of depth $\leq \log(2m)$, hence as a **CNF**/**DNF** formula of bottom fan-in at most $\log(2m)$. This means that applying the restriction $\rho \circ \tau_\ell$, the circuit f collapses to a depth $d-1$ Boolean circuit with bottom fan-in $t' \leq \log(2m)$ and effective size at most m.[5] By the induction hypothesis, for any k' we have $\mathbf{W}^{\geq k'}[f|_\rho|_{\tau_\ell}] \leq 8^{d-2} \cdot 2^{-\Omega(k'/(t' \log^{d-3}(2m)))}$. Setting $k' = \lfloor kp \rfloor - D \geq \lfloor kp/2 \rfloor \geq \frac{k}{96t} - 1$ and applying Lemma 13 we have

$$\mathbf{W}^{\geq \lfloor kp \rfloor}[f|_\rho] \leq \max_\ell \mathbf{W}^{\geq k'}[f|_\rho|_{\tau_\ell}] \leq 8^{d-2} \cdot 2^{-k'/(20t' \cdot (96 \log(2m))^{d-3})} \leq 8^{d-2} \cdot 2^{1-k/\left(20t(96\log(2m))^{d-2}\right)},$$

and

$$\mathbf{W}^{\geq k}[f] \leq 4 \cdot 8^{d-2} \cdot 2^{-k/\left(20t(96\log(2m))^{d-2}\right)} + 4 \cdot 2^{-k/(96t\log(2m))} \leq 8^{d-1} \cdot 2^{-k/\left(20t(96\log(2m))^{d-2}\right)}.$$ ◀

▶ **Theorem 18** (Theorem 1, restated)**.** *Let f be an Boolean circuit of depth d and size $m > 1$. Then, $\mathbf{W}^{\geq k}[f] \leq 2 \cdot 2^{-k/(c_d \log^{d-1}(m))}$ where $c_d = 60d \cdot 192^{d-1} \leq 216^d$. Equivalently, $\mathbf{W}^{\geq k}[f] \leq 2 \cdot e^{-k/(c'_d \log^{d-1}(m))}$ where $c'_d = \log_2(e) \cdot 60d \cdot 192^{d-1} \leq 2 \cdot 216^d$.*

Proof. Let f be a function computed by a Boolean circuit of depth d and m gates. We add a dummy layer of fan-in 1 gates in between the inputs and the layer next to them. Thus, f is realized by an Boolean circuit of depth $d+1$, effective size m and bottom fan-in 1. Plugging this into Theorem 16 gives $\mathbf{W}^{\geq k}[f] \leq 2^{3d - k/(20 \cdot 96^{d-1} \cdot \log^{d-1}(2m))}$. Hence, by Claim 17, we get

$$\mathbf{W}^{\geq k}[f] \leq 2 \cdot 2^{-k/(3d \cdot 20 \cdot 96^{d-1} \cdot \log^{d-1}(2m))} \leq 2 \cdot 2^{-k/(60d \cdot 96^{d-1} \cdot 2^{d-1} \cdot \log^{d-1}(m))},$$

where we used $\log(2m) \leq 2\log(m)$ for $m > 1$. ◀

4 Applications to Pseudorandomness

4.1 Improving Braverman's Analysis

▶ **Definition 19.** Denote by $\text{tail}(m, d, k)$ the maximal $\mathbf{W}^{\geq k}[F]$ over all Boolean circuits F of size $\leq m$ and depth $\leq d$.

By Theorem 18, $\text{tail}(m, d, k) \leq 2 \cdot 2^{-k/(c_d \log^{d-1}(m))}$. Braverman's Theorem can be rephrased as follows (we show that this is indeed the case in Appendix B).

▶ **Theorem 20** ([7])**.** *Let $s_1, s_2 \geq \log m$ be any parameters. Let F be a Boolean function computed by a circuit of depth d and size m. Let μ be an r-independent distribution where*

$$r = r(s_1, s_2, d) = 2((s_1 \cdot \log m)^d + s_2)$$

[5] We only introduce new gates with distance 1 from the inputs – which does not increase the effective size.

then

$$|\mathop{\boldsymbol{E}}_{\mu}[F] - E[F]| < \varepsilon(s_1, s_2, d),$$

where $\varepsilon(s_1, s_2, d) = 0.82^{s_1} \cdot (6m) + 2^{4(s_1 \cdot \log m)^d \log m} \cdot \mathrm{tail}(m^3, d+3, s_2)$

Picking $s_1 := 5\log(12m/\varepsilon)$ and $s_2 := \left(c_{d+3}\log(m^3)^{d+2}\right) \cdot 8 \cdot (s_1 \cdot \log m)^d \cdot \log(m)$ we get the following corollary.

▶ **Theorem 21.** *$r(m, d, \varepsilon)$-independence ε-fools Boolean circuits of depth d and size m, where*

$$\begin{aligned} r(m,d,\varepsilon) &= 2((s_1 \cdot \log m)^d + s_2) \le 4s_2 \\ &= 32 \cdot c_{d+3} \cdot (5\log(12m/\varepsilon))^d \cdot 3^{d+2} \cdot (\log m)^{2d+3} \\ &\le O(\log(m/\varepsilon))^d \cdot (\log m)^{2d+3} . \end{aligned}$$

4.2 Improving Bazzi's Analysis

Bazzi [4] showed that $O(\log^2(m/\varepsilon))$ independence ε-fools **DNFs** of size m. We show that $O(\log(m/\varepsilon) \cdot \log(m))$ independence suffices. For $\varepsilon \le 1/m^{\Omega(1)}$ this bound is tight, due to the example of Mansour from [23].

▶ **Theorem 22** ([4], [30]). *Let F be a* **DNF** *with m terms, and t be some parameter. Then, F is $m^3 \cdot \mathrm{tail}(m, 2, (k-3t)/2) + m2^{-t}$ fooled by any k-wise independence.*

Picking $t := \log(2m/\varepsilon)$ and $k := 3t + 2c_2\log(m)\log(4m^3/\varepsilon) = O(\log(m)\log(m/\varepsilon))$, we get that k-wise independence ε-fools **DNFs** with m terms since

$$m^3 \cdot \mathrm{tail}(2, m, (k-3t)/2) + m2^{-t} \le m^3 \cdot 2 \cdot 2^{\frac{-c_2\log(m)\log(4m^3/\varepsilon)}{c_2\log(m)}} + \frac{\varepsilon}{2} \le \varepsilon .$$

5 On Fourier Concentration, Switching Lemmas and Influence Moments

In this section, we connect different notions of Fourier concentration of Boolean functions. We begin by introducing some new definitions, and then move to state and prove the connections between the different notions. We end this Section, with the proof of Theorem 2, which is a result of Theorem 1 and the connections established in this section.

5.1 Influence Moments

In this section we introduce derivatives and influences of sets of variables. A different definition to the influence of a set was made in [20]. There, the influence of a set J was defined to be the probability that under a uniform restriction of J^c to constants, the function's value is still undetermined. We choose a different variant, which has a much nicer Fourier expression.

We start with the standard definition of discrete derivatives and influences of Boolean functions.

▶ **Definition 23** (Discrete Derivative, Influence). Let $f\colon \{-1,1\}^n \to \mathbb{R}$ and $i \in [n]$. The i-th discrete derivative operator D_i maps the function f to the function $D_i f : \{-1,1\}^n \to \mathbb{R}$ defined by

$$D_i f(x) = \frac{f(x^{(i\mapsto 1)}) - f(x^{(i\mapsto -1)})}{2} .$$

where $x^{(i\mapsto b)} = (x_1, \ldots, x_{i-1}, b, x_{i+1}, \ldots, x_n)$. The influence of coordinate i on f is defined as

$$\mathrm{Inf}_i(f) = \mathop{\mathbf{E}}_x[(D_i f(x))^2] \,.$$

The generalization to sets of more than one variable is the following.

▶ **Definition 24** (Discrete Derivative and Influence of a Set)**.** Let $f\colon \{-1,1\}^n \to \mathbb{R}$ and $T \subseteq [n]$, and write $T = \{j_1, \ldots, j_k\}$. The T-th (discrete) derivative operator, D_T, maps the function f to the function $D_T f : \{-1,1\}^n \to \mathbb{R}$ defined by

$$D_T f(x) = D_{j_1} D_{j_2} \ldots D_{j_k} f(x) \,.$$

The influence of subset T on f is defined as

$$\mathrm{Inf}_T(f) = \mathop{\mathbf{E}}_x\left[(D_T f(x))^2\right] \,.$$

The following claim gives equivalent formulations for the function $D_T f$ (and also implies that D_T is well defined, i.e., that $D_T f$ does not depend on the order of indices in T).

▶ **Claim 25.**

$$D_T f(x) = \frac{1}{2^{|T|}} \sum_{z \in \{-1,1\}^T} f(x^{(T\mapsto z)}) \cdot \prod_{i \in T} z_i = \sum_{S \supseteq T} \hat{f}(S) \cdot \prod_{i \in S \setminus T} x_i$$

where $x^{(T\mapsto z)}$ is the vector in $\{-1,1\}^n$ whose i-th coordinate equals z_i whenever $i \in T$, and equals x_i otherwise.

The proof uses a straightforward inductive argument, and is given for completeness in Appendix A. Note that if $f : \{-1,1\}^n \to \{-1,1\}$, then the T-th derivative of f is $2^{-|T|}$ granular (i.e., $D_T f(x)$ is an integer times $2^{-|T|}$), since $D_T f(x)$ is a sum of integers divided by $2^{|T|}$. The following claim follows from Parseval's identity and the previous claim.

▶ **Claim 26.** $\mathrm{Inf}_T(f) = \sum_{S \supseteq T} \hat{f}(S)^2$.

▶ **Definition 27** (Total Degree-k Influence)**.** The total degree-k influence is defined as

$$\mathrm{Inf}^k(f) \triangleq \sum_{T : |T| = k} \mathrm{Inf}_T(f) \,.$$

Claim 26 gives the following Fourier expression for the total degree-k influence:

$$\mathrm{Inf}^k(f) = \sum_{S : |S| \geq k} \hat{f}(S)^2 \cdot \binom{|S|}{k} = \sum_{d \geq k} \mathbf{W}^d[f] \cdot \binom{d}{k} \,. \tag{2}$$

We state the following simple lemma expressing $\mathrm{Inf}^k(f)$ in terms of $\mathbf{W}^{\geq d}[f]$ instead of $\mathbf{W}^d[f]$.

▶ **Lemma 28.** $\mathrm{Inf}^k(f) = \sum_{d \geq k} \mathbf{W}^{\geq d}[f] \cdot \binom{d-1}{k-1}$ *for all* $k \in \mathbb{N}$.

Proof. We perform some algebraic manipulations on Equation (2):

$$\begin{aligned}
\mathrm{Inf}^k(f) = \sum_{d \geq k} \mathbf{W}^d[f] \cdot \binom{d}{k} &= \sum_{d \geq k} \left(\mathbf{W}^{\geq d}[f] - \mathbf{W}^{\geq d+1}[f]\right) \cdot \binom{d}{k} \\
&= \mathbf{W}^{\geq k}[f] + \sum_{d \geq k+1} \mathbf{W}^{\geq d}[f] \cdot \left(\binom{d}{k} - \binom{d-1}{k}\right) \\
&= \mathbf{W}^{\geq k}[f] + \sum_{d \geq k+1} \mathbf{W}^{\geq d}[f] \cdot \binom{d-1}{k-1} \\
&= \sum_{d \geq k} \mathbf{W}^{\geq d}[f] \cdot \binom{d-1}{k-1}
\end{aligned}$$

◀

5.2 Connections between Four Fourier Concentration Properties

In this section we show connections between four attributes of Boolean functions, and establish equivalence between three of them. The properties, each relative to a parameter t, are the following:

- **ESFT**: Exponentially small Fourier tails.

  $$\forall k : \mathbf{W}^{\geq k}[f] \leq e^{-\Omega(k/t)} .$$

- **SLTP**: Switching lemma type property / degree shrinkage

  $$\forall d, p : \Pr_{\rho \sim \mathcal{R}_p} [\deg(f|_\rho) = d] \leq O(pt)^d .$$

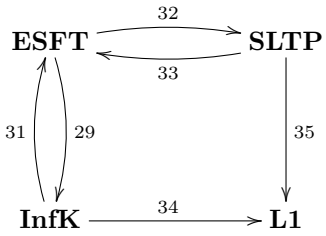
- **L1**: Bounded spectral norm of the k-th level.

  $$\forall k : \sum_{|S|=k} |\hat{f}(S)| \leq O(t)^k .$$

- **InfK**: Bounded total degree-k influence.

  $$\forall k : \mathrm{Inf}^k[f] \leq O(t)^k .$$

In Lemmata 29, 31, 32, 33, 34, 35, we show the following connections:



We remark that Lemma 35 is due to Mansour [25], and Lemma 33 is due to Linial et al. [22]. Note that **L1** does not imply any other property, because one can take for example the parity function, which has the **L1** property with $t = 1$. However, this function has very large Fourier tails, very high degree under random restriction, and $\binom{n}{k}$ total degree-k influence. Anything that implies **SLTP** and **L1** needs f to be Boolean. Other relations generalize to bounded real-valued functions.

In the remainder of this section we state Lemmata 29, 31, 32, 33, 34, 35 more accurately and prove them.

▶ **Lemma 29.** *Let $t > 0, C > 0$. If $\mathbf{W}^{\geq d}[f] \leq C \cdot e^{-d/t}$ for all d, then $\mathrm{Inf}^k[f] \leq C \cdot t^k$ for all k.*

In the proof of Lemma 29, we use the following simple fact that follows from Newton's generalized binomial theorem.

▶ **Fact 30.** *Let $|x| < 1$, and $k \in \mathbb{N}$. Then, $\sum_{d=k}^{\infty} \binom{d-1}{k-1} \cdot x^d = \frac{x^k}{(1-x)^k}$.*

Proof of Lemma 29. We shall prove for $C = 1$, the proof generalizes for all C. Denote $a := e^{-1/t}$. Using Lemma 28 we bound the total degree-k influence:

$$\mathrm{Inf}^k(f) = \sum_{d \geq k} \mathbf{W}^{\geq d}[f] \cdot \binom{d-1}{k-1} \leq \sum_{d \geq k} e^{-d/t} \cdot \binom{d-1}{k-1} = \sum_{d \geq k} a^d \cdot \binom{d-1}{k-1}$$

Using Fact 30 with $x := a$ gives

$$\mathrm{Inf}^k(f) \leq \frac{a^k}{(1-a)^k} = \frac{1}{(1/a-1)^k} = \frac{1}{\left(e^{1/t}-1\right)^k} \leq \frac{1}{(1/t)^k} = t^k$$

where in the last inequality we used the fact that $e^x - 1 \geq x$ for all $x \in \mathbb{R}$. ◀

The reverse relation holds too, i.e. **InfK** implies **ESFT**.

▶ **Lemma 31.** *Let $t > 0, C > 0$. If $\mathrm{Inf}^k[f] \leq C \cdot t^k$ for all k, then $\mathbf{W}^{\geq d}[f] \leq C \cdot e \cdot t \cdot e^{-(d-1)/et}$ for all d.*

Proof. We shall prove for $C = 1$, the proof generalizes for all C. By Lemma 28, $\mathbf{W}^{\geq d}[f] \cdot \binom{d-1}{k-1} \leq \mathrm{Inf}^k[f] \leq t^k$. Hence $\mathbf{W}^{\geq d}[f] \leq t^k / \binom{d-1}{k-1}$. We can pick any k to optimize this bound. Picking $k = \lfloor (d-1)/et \rfloor + 1$ we get

$$\mathbf{W}^{\geq d}[f] \leq t^k / \left(\frac{d-1}{k-1}\right)^{k-1} \leq t \cdot e^{-(k-1)} \leq e \cdot t \cdot e^{-(d-1)/et} \,. \quad ◀$$

In our previous work [33], the following relation (**ESFT** implies **SLTP**) was established.

▶ **Lemma 32** ([33]). *Let $t, C > 0$, and $f : \{-1,1\}^n \to \{-1,1\}$. If $\mathbf{W}^{\geq k}[f] \leq C \cdot e^{-k/t}$ for all k, then $\boldsymbol{Pr}_{\rho \sim \mathcal{R}_p}[\deg(f|_\rho) = d] \leq C \cdot (4pt)^d$ for all p, d.*

We give a slightly shorter proof, using the total degree-d influence.

Proof. We shall prove for $C = 1$, the proof generalizes for all C. The proof goes by showing that

$$\mathop{\mathbf{E}}_{\rho}[\mathbf{W}^d[f|_\rho]] \leq (pt)^d \tag{3}$$

and

$$\mathop{\mathbf{E}}_{\rho}[\mathbf{W}^d[f|_\rho]] \geq 4^{-d} \cdot \mathop{\mathbf{Pr}}_{\rho}[\deg(f|_\rho) = d] \,. \tag{4}$$

Equation (4) is true since

$$\mathop{\mathbf{E}}_{\rho}[\mathbf{W}^d[f|_\rho]] \geq \mathop{\mathbf{E}}_{\rho}[\mathbf{W}^d[f|_\rho] | \deg(f|_\rho) = d] \cdot \mathop{\mathbf{Pr}}_{\rho}[\deg(f|_\rho) = d] \,.$$

and the (random) Boolean function $f|_\rho$ has Fourier mass at least 4^{-d} if $\deg(f|_\rho) = d$, by the granularity of low degree functions – Fact 9.

We are left to prove Equation (3). Using Fact 7, we have

$$\mathop{\mathbf{E}}_{\rho}[\mathbf{W}^d[f|_\rho]] = \sum_{k=d}^{n} \mathbf{W}^k[f] \binom{k}{d} p^d (1-p)^{k-d} \leq p^d \sum_{k=d}^{n} \mathbf{W}^k[f] \binom{k}{d} = p^d \cdot \mathrm{Inf}^d[f] \leq (pt)^d \,,$$

where in the last inequality we used Lemma 29. ◀

Linial, Mansour and Nisan [22] proved that **SLTP** implies **ESFT**.

▶ **Lemma 33** ([22], restated slightly). *Let $t > 0, C > 0$, and $f : \{-1,1\}^n \to [-1,1]$. If for all $d \in \mathbb{N}, p \in (0,1)$, $\boldsymbol{Pr}_{\rho \sim \mathcal{R}_p}[\deg(f|_\rho) \geq d] \leq C\,(tp)^d$, then for any k, $\mathbf{W}^{\geq k}[f] \leq 2e \cdot C \cdot e^{-k/te}$.*

The proof is given in [22]; we give it here for completeness.

Proof. Pick $p = 1/et$, then by Lemma 10, and the fact that $\mathbf{W}^{\geq \lfloor kp \rfloor}[f|_\rho]$ is always at most 1 and equals 0 whenever $\deg(f|_\rho) < \lfloor kp \rfloor$, we get

$$\mathbf{W}^{\geq k}[f] \leq 2 \mathop{\mathbf{E}}_{\rho} \left[\mathbf{W}^{\geq \lfloor kp \rfloor}[f|_\rho] \right] \leq 2 \mathop{\mathbf{E}}_{\rho} [\mathbf{Pr}[\deg(f|_\rho) \geq \lfloor kp \rfloor]] \leq 2C(1/e)^{\lfloor k/et \rfloor} . \qquad \blacktriangleleft$$

The next lemma proves that **InfK** implies **L1**.

▶ **Lemma 34.** *If f is Boolean, then $L_{1,k}[f] \leq 2^k \cdot \mathrm{Inf}^k[f]$.*

Proof. It is easy to see from Claim 25 that for any subset $T \subseteq [n]$,

$$\mathop{\mathbf{E}}_{x}[D_T f(x)] = \mathop{\mathbf{E}}_{x} \left[\sum_{S \supseteq T} \hat{f}(S) \prod_{i \in S \setminus T} x_i \right] = \hat{f}(T) .$$

Recall that if f is Boolean, then $D_T f(x)$ is $2^{-|T|}$ granular, which implies that $\forall x : |D_T f(x)| \leq 2^{|T|}(D_T f(x))^2$. Hence,

$$|\hat{f}(T)| = |\mathop{\mathbf{E}}_{x}[D_T f(x)]| \leq \mathop{\mathbf{E}}_{x}[|D_T f(x)|] \leq 2^{|T|} \mathop{\mathbf{E}}_{x}[(D_T f(x))^2] = 2^{|T|}\mathrm{Inf}_T(f) .$$

Summing over all sets T of size k completes the proof. $\blacktriangleleft$

Remark: It is necessary that f is Boolean in Lemma 34, since otherwise we can have the function

$$f_{t,k}(x) = \sum_{S \subseteq [n], |S|=k} \frac{1}{\sqrt{\binom{n}{k}} e^{k/2t}} \prod_{i \in S} x_i$$

which maps $\{-1,1\}^n$ to $\mathbb{R}$, has $\mathbf{W}^{\geq k}[f_{t,k}] = \mathbf{W}^k[f_{t,k}] = e^{-k/t}$, and $\mathrm{Inf}^k[f_t] \leq t^k$, but

$$L_{1,k}[f_t] = \sqrt{\binom{n}{k}} e^{-k/2t} \geq \left(\frac{n}{ke^{1/t}} \right)^{k/2}$$

is much larger than $O(t)^k$ for $n = \omega(kt^2 e^{1/t})$.

Next, Mansour [25] proved that **SLTP** implies **L1**.

▶ **Lemma 35** ([25])**.** *Let $t > 0$, and $f : \{-1,1\}^n \to \{-1,1\}$. If for all d, p, $\boldsymbol{Pr}_{\rho \sim \mathcal{R}_p}[\deg(f|_\rho) = d] \leq C(pt)^d$, then $\forall k : L_{1,k}[f] \leq 2C(4t)^k$.*

Proof. We shall prove for $C = 1$, the proof generalizes for all C. We first prove that for any function $f : \{-1,1\}^n \to \mathbb{R}$, $p \in [0,1], k \in \mathbb{N}$ we have $L_{1,k}(f) \leq \frac{1}{p^k} \mathbf{E}_{\rho \sim \mathcal{R}_p}[L_{1,k}[f|_\rho]]$.

$$\begin{aligned} L_{1,k}[f] &= \sum_{S:|S|=k} |\hat{f}(S)| = \sum_{S:|S|=k} \left| \frac{1}{p^k} \mathop{\mathbf{E}}_{\rho \sim \mathcal{R}_p} \left[\widehat{f|_\rho}(S) \right] \right| && \text{(Fact 6)} \\ &\leq \sum_{S:|S|=k} \frac{1}{p^k} \mathop{\mathbf{E}}_{\rho \sim \mathcal{R}_p} \left[|\widehat{f|_\rho}(S)| \right] = \frac{1}{p^k} \mathop{\mathbf{E}}_{\rho \sim \mathcal{R}_p} \left[\sum_{S:|S|=k} |\widehat{f|_\rho}(S)| \right] \\ &= \frac{1}{p^k} \mathop{\mathbf{E}}_{\rho \sim \mathcal{R}_p} [L_{1,k}[f|_\rho]] . && (5) \end{aligned}$$

Next, we show that for $f : \{-1,1\}^n \to \{-1,1\}$, if there exists $t > 0$ such that for all d, p, $\mathbf{Pr}[\deg(f|_\rho) = d] \leq (pt)^d$, then $\mathbf{E}_{\rho\sim\mathcal{R}_p}[L_1[f|_\rho]] \leq 2$ for $p = 1/4t$. Conditioning on $\deg(f|_\rho) = d$ and using Fact 9, we have $L_1[f|_\rho] \leq 2^d$. Hence,

$$\mathop{\mathbf{E}}_{\rho\sim\mathcal{R}_p}[L_1[f|_\rho]] = \sum_{d=0}^{n} \mathop{\mathbf{E}}_{\rho\sim\mathcal{R}_p}[L_1[f|_\rho]|\deg(f|_\rho) = d] \cdot \mathbf{Pr}[\deg(f|_\rho) = d] \leq \sum_{d=0}^{n} 2^d \cdot \left(\tfrac{1}{4}\right)^d \leq 2\,. \quad (6)$$

Plugging Equation (6) in Equation (5) with $p = 1/4t$ we get

$$L_{1,k}[f] \leq \frac{1}{p^k} \mathop{\mathbf{E}}_{\rho\sim\mathcal{R}_p}[L_{1,k}[f|_\rho]] \leq \frac{1}{p^k} \mathop{\mathbf{E}}_{\rho\sim\mathcal{R}_p}[L_1[f|_\rho]] \leq (4t)^k \cdot 2\,. \qquad \blacktriangleleft$$

The next lemma is relevant to the learnability results given in [25] and [22].

▶ **Lemma 36.** *Let f be a Boolean function, let $t \geq 1$ and C be some positive constant. If $\mathbf{W}^{\geq k}[f] \leq C \cdot e^{-k/t}$ for all k, then f is ε-concentrated on at most $t^{O(t\log(1/\varepsilon))}$ Fourier coefficients.*

Here, by ε-concentrated on r coefficients we mean that there exist r subsets of $[n]$, $\{S_1, \ldots, S_r\}$, which captures $1-\varepsilon$ of the Fourier mass of f, i.e. $\sum_{i=1}^{r} \hat{f}(S_i)^2 \geq 1-\varepsilon$.

Proof. We shall prove for $C = 1$, the proof generalizes for all constant C. Let $w := t \cdot \ln(2/\varepsilon)$. First it is enough to consider Fourier coefficients of sets of size $\leq w$, since the sum of squares of Fourier coefficients of larger sets is at most $\varepsilon/2$. Now $\sum_{S:|S|\leq w} |\hat{f}(S)| = \sum_{i=0}^{w} L_{1,i}[f]$. Using Lemmata 29 and 34 we get

$$\sum_{i=0}^{w} L_{1,i}[f] \leq \sum_{i=0}^{w} 2^i t^i \underset{(t\geq 1)}{\leq} t^w 2^{w+1}\,.$$

Letting $\mathcal{F} = \{S : |S| \leq w, |\hat{f}(S)| \geq \frac{\varepsilon/2}{t^w 2^{w+1}}\}$ we get by Parseval's identity that

$$\sum_{S\in\mathcal{F}} \hat{f}(S)^2 = 1 - \sum_{|S|>w} \hat{f}(S)^2 - \sum_{|S|\leq w, S\notin\mathcal{F}} \hat{f}(S)^2\,,$$

where we already noted that $\sum_{|S|>w} \hat{f}(S)^2 \leq \varepsilon/2$. To bound the last term

$$\sum_{|S|\leq w, S\notin\mathcal{F}} \hat{f}(S)^2 \leq \max\{|\hat{f}(S)| : |S| \leq w, S \notin \mathcal{F}\} \cdot \sum_{|S|\leq w} |\hat{f}(S)| \leq \varepsilon/2\,.$$

Hence, $\sum_{S\in\mathcal{F}} \hat{f}(S)^2 \geq 1-\varepsilon$. It remain to figure out the size of $\mathcal{F}$. Since every coefficient in $\mathcal{F}$ contributes at least $\frac{\varepsilon/2}{t^w 2^{w+1}}$ to the sum $\sum_{i=0}^{w} L_{1,i}[k]$, and this sum is at most $t^w 2^{w+1}$ we get that the size of $\mathcal{F}$ is at most $2(t^w 2^{w+1})^2/\varepsilon = O(t)^{2t\ln(1/\varepsilon)}$, which completes the proof. $\blacktriangleleft$

5.3 Theorem 2

Immediate from Theorem 18, Lemmata 29, 32, 34, and 36 we get the following corollary.

▶ **Theorem 37** (Thm. 2, restated)**.** *Let f be a Boolean circuit of depth d and size $m > 1$. Then,*

1. *For all k, p, $\boldsymbol{Pr}_{\rho\sim\mathcal{R}_p}[\deg(f|_\rho) = k] \leq 2 \cdot (4p \cdot c'_d \log^{d-1}(m))^k$.*
2. *For all k, $\mathrm{Inf}^k[f] \leq 2 \cdot (c'_d \log^{d-1}(m))^k$.*
3. *For all k, $L_{1,k}[f] = \sum_{S:|S|=k} |\hat{f}(S)| \leq 2 \cdot (2c'_d \log^{d-1}(m))^k$.*
4. *f is ε-concentrated on at most $O(\log^{d-1} m)^{O(\log^{d-1}(m)\log(1/\varepsilon))} = 2^{O(\log\log(m)\log^{d-1}(m)\log(1/\varepsilon))}$ Fourier coefficients.*

6 Short Proofs for Known Results

In this section, we give simple proofs for two known results based on Theorem 37.

6.1 Bounded-Depth Circuits Cannot Approximate Majority

The next result states that nearly balanced symmetric functions, and in particular the Majority function, cannot be well approximated by a small and shallow circuit.

▶ **Theorem 38.** *Let $g : \{-1,1\}^n \to \{-1,1\}$ be a symmetric function on n variables. Let $f : \{-1,1\}^n \to \{-1,1\}$ be depth d size m circuit, and assume that*

$$c'_d \log^{d-1}(m) \leq (n/100\ln(n))^{1/3} .$$

Then,

$$\mathrm{Cor}(f,g) \triangleq \left|\mathop{\boldsymbol{E}}_{x}[f(x)g(x)]\right| \leq |\hat{g}(\emptyset)| + \frac{\sqrt{2}+8c'_d\log^{d-1}(m)}{\sqrt{n}}$$

Proof. Since g is a symmetric Boolean function, for all $S \subseteq [n]$, $\hat{g}(S)^2 \cdot \binom{n}{|S|} = \sum_{T:|T|=|S|} \hat{g}(T)^2 \leq 1$. Hence, $|\hat{g}(S)| \leq \frac{1}{\sqrt{\binom{n}{|S|}}}$. Let ℓ be some parameter we shall set later. Then,

$$\left|\mathop{\mathbf{E}}_{x}[f(x)g(x)]\right| \leq \sum_{S} |\hat{f}(S)\hat{g}(S)| = |\hat{f}(\emptyset)\hat{g}(\emptyset)| + \sum_{k=1}^{\ell}\sum_{S:|S|=k} |\hat{f}(S)\hat{g}(S)| + \sum_{S:|S|>\ell} |\hat{f}(S)\hat{g}(S)|. \tag{7}$$

We bound each of the three terms in the RHS of Equation (7). The first term is at most $|\hat{g}(\emptyset)|$. For the third term we use Cauchy-Schwartz, Theorem 18, and Parseval's identity ($\sum_{S:|S|>\ell}\hat{g}(S)^2 \leq 1$), to get

$$\sum_{S:|S|>\ell} |\hat{f}(S)\hat{g}(S)| \leq \sqrt{\sum_{S:|S|>\ell}\hat{f}(S)^2 \sum_{S:|S|>\ell}\hat{g}(S)^2} \leq \sqrt{2\cdot e^{-\ell/\left(c'_d\log^{d-1}(m)\right)}} .$$

Picking $\ell := \ln(n)\cdot c'_d\log^{d-1}(m)$ this is smaller than $\sqrt{2/n}$. For the second term in the RHS of Equation (7), we use the estimates on $L_{1,k}(f)$ and $|\hat{g}(S)|$, to get

$$\sum_{S:|S|=k} |\hat{g}(S)\hat{f}(S)| \leq \frac{1}{\sqrt{\binom{n}{k}}}\cdot\sum_{S:|S|=k}|\hat{f}(S)| \leq \frac{2\cdot(2c'_d\log^{d-1}(m))^k}{\sqrt{\binom{n}{k}}} \leq 2\cdot\left(\frac{2c'_d\log^{d-1}(m)}{\sqrt{n/k}}\right)^k .$$

We denote by $D_k := 2\cdot\left(\frac{2c'_d\log^{d-1}(m)}{\sqrt{n/k}}\right)^k$. The ratio between two consecutive terms D_{k+1}/D_k for $k+1 \leq \ell$ is at most

$$\frac{2c'_d\log^{d-1}(m)}{\sqrt{n}}\sqrt{\frac{(k+1)^{k+1}}{k^k}} \leq \frac{2c'_d\log^{d-1}(m)}{\sqrt{n}}\sqrt{e\cdot(k+1)} \leq \frac{2c'_d\log^{d-1}(m)}{\sqrt{n}}\sqrt{e\cdot\ell} \leq \frac{1}{2} ,$$

where we used the choice of ℓ and the assumption $c'_d\log^{d-1}(m) \leq \left(\frac{n}{100\ln n}\right)^{1/3}$ for the last inequality to hold. We get that the sum $\sum_{1\leq|S|\leq\ell}|\hat{f}(S)\hat{g}(S)|$ is at most $D_1+D_2+\ldots+D_\ell \leq 2D_1$. Overall, we get

$$\mathop{\mathbf{E}}_{x}[f(x)g(x)] \leq |\hat{g}(\emptyset)| + \frac{\sqrt{2}+8c'_d\log^{d-1}(m)}{\sqrt{n}} . \qquad ◀$$

We remark that although our proof is Fourier analytical, it differs from the standard argument that is used to bound the correlation of bounded depth circuits with parity for example. The standard argument shows that two functions are $o(1)$ correlated by proving that one is $1-o(1)$ concentrated on the low levels of the Fourier spectrum while the other is $1-o(1)$ concentrated on the high levels. Here, however, if we take g to be the Majority function, and f to be an $\mathbf{AC^0}$ circuit, then both f and g are 0.99-concentrated on the first $O(\text{poly}\log(n))$ levels of their Fourier spectrum. We deduce the small correlation by showing that f must be very imbalanced on those levels, which is captured by having small $L_{1,k}$ norm. In contrast, the Majority function is symmetric – its Fourier mass on level k is equally spread on the different coefficients. Combining these two properties guarantees small correlation.

6.2 The Coin-Problem

▶ **Theorem 39.** *Let $f:\{-1,1\}^n\to\{-1,1\}$ be a depth d size m circuit, and let $p\in[0,1]$. Then, f distinguishes between unbiased coins and coins with bias p with advantage at most $6c'_d p\log^{d-1}(m)$.*

Proof. We can assume $pc'_d\log^{d-1}(m)\leq 1/6$, since otherwise the result is trivial. For $-1\leq p\leq 1$, a p-biased coin is a random variable which gets 1 with probability $(1+p)/2$ and -1 with probability $(1-p)/2$, i.e., this is a biased coin whose expectation is p. Let U_n be the distribution of n independent 0-biased coins, and $B(n,p)$ be the distribution of n independent p-biased coins. We have

$$\begin{aligned}\text{Distinguishability}(f) &\triangleq \left|\mathop{\mathbf{E}}_{x\sim U_n}[f(x)]-\mathop{\mathbf{E}}_{x\sim B(p,n)}[f(x)]\right| = \left|\hat{f}(\emptyset)-\sum_{S\subseteq[n]}\hat{f}(S)p^{|S|}\right| \\ &= \left|\sum_{S\neq\emptyset}\hat{f}(S)p^{|S|}\right| \leq \sum_{k=1}^{n} p^k\cdot 2\cdot\left(2c'_d\log^{d-1}(m)\right)^k \\ &\leq 2p\cdot\left(2c'_d\log^{d-1}(m)\right)\cdot\sum_{k=1}^{\infty}(1/3)^{k-1} = 3p\cdot\left(2c'_d\log^{d-1}(m)\right). \quad ◀\end{aligned}$$

7 A New Proof for Håstad's Switch-Many Lemma

In this section, we give a new proof for Håstad's [16] Switch-Many Lemma, i.e., Lemma 15. The new proof follows Razborov's [29] approach, and its recent simplification by Thapen [34] for Håstad's original switching lemma [14].

Notation. We denote by $\mathcal{R}$ the set of all restrictions on n variables. For a sequence of indices $S\in[n]^k$ with no repetitions, and a string $\sigma\in\{0,1\}^k$ we denote by $(S\to\sigma)$ the restriction which fixes S_i to σ_i for $i\in[k]$ and leaves all other variables free. For two restrictions ρ,σ we denote by $\rho\sigma$ their composition. For a sequence $S\in\Sigma^k$ over some alphabet Σ, and two indices i and j such that $1\leq i\leq j\leq k$, we denote by $S[i:j]$ the subsequence $(S_i,\ldots,S_j)$, and by $S[i]$ the element S_i.

7.1 The Canonical Decision Tree

Let F be an r-**DNF**, i.e., an OR of ANDs where each AND has at most r input literals from $x_1,\ldots,x_n,\neg x_1,\ldots,\neg x_n$. Let ρ be a restriction. The canonical tree $T(F,\rho)$ is defined by the following decision procedure: Look through F for the first term C_1, such that $C_1|_\rho\not\equiv 0$. If no

such term exists, then halt and output 0. Otherwise, let A be the set of free variables in C_1 under ρ. Query the variables in A and let $\pi_1, \ldots, \pi_{|A|}$ be their assignment. If the term C_1 is satisfied under the assignment (in particular if $A = \emptyset$), then halt and output 1 . Otherwise, repeat the process with $\rho(A \to (\pi_1, \pi_2, \ldots, \pi_{|A|}))$ instead of ρ. We keep iterating until one of the aforementioned halting conditions hold.

7.2 Restriction Tree for Multiple DNFs

Let $F_1, \ldots, F_m$ be r-**DNFs**. We define the d-restriction-tree complexity of $F_1, \ldots, F_m$ to be the minimal depth of a restriction tree such that under the restriction defined by each leaf, each **DNF** F_i is of canonical decision-tree-complexity at most d. We denote this complexity by $\mathrm{RT}_d(\{F_1, \ldots, F_m\})$.

▶ **Theorem 40.**

$$\Pr_{\rho \sim \mathcal{R}_p}[\mathrm{RT}_d(\{F_1|_\rho, \ldots, F_m|_\rho\}) \geq k] \leq m^{\lceil k/(d+1) \rceil} \cdot \left(\frac{24pr}{1-p}\right)^k .$$

The following is a corollary of Theorem 40.

▶ **Corollary 41.** *Let $F_1, \ldots, F_m$ be r-**DNFs**. Let k, d be positive integers, $0 \leq p \leq 1$, and assume $2^{d+1} \geq m$. Then,*

$$\Pr_{\rho \sim \mathcal{R}_p}[\mathrm{RT}_d(\{F_1|_\rho, \ldots, F_m|_\rho\}) \geq k] \leq m \cdot (49pr)^k . \tag{8}$$

Proof. We can assume without loss of generality that $p < 1/49$ since otherwise the RHS of Eq. (8) is at least 1 and the LHS is always at most 1. We get

$$\begin{aligned}
\Pr_{\rho \sim \mathcal{R}_p}[\mathrm{RT}_d(\{F_1|_\rho, \ldots, F_m|_\rho\}) \geq k] &\leq m^{\lceil k/(d+1) \rceil} \cdot (24pr/(1-p))^k && \text{(Theorem 40)} \\
&\leq m^{1+k/(d+1)} \cdot (24pr/(1-p))^k \\
&= m \cdot \left(\frac{m^{1/(d+1)} \cdot 24pr}{1-p}\right)^k \\
&\leq m \cdot \left(\frac{2 \cdot 24pr}{1-p}\right)^k && (m \leq 2^{d+1}) \\
&\leq m \cdot (49pr)^k . && (1-p > 48/49)
\end{aligned}$$

◀

We will prove Theorem 40 based on the approach of Thapen [34] which simplified Razborov's [29] and Beame's [5] proofs for the (original) switching lemma. The idea of the proof is that in order to show that some event A happens with low probability, it is sufficient to show that there exists some other event B (not necessarily disjoint of A) that happens with probability much larger than A. For example, if $\mathbf{Pr}[B] \geq M \cdot \mathbf{Pr}[A]$ (think of M as some large factor) then since $\mathbf{Pr}[B] \leq 1$ it means that $\mathbf{Pr}[A] \leq 1/M$.

The following is the main lemma in this section, from which we deduce Theorem 40 quite easily.

▶ **Lemma 42.** *Let S be the set of restrictions under which $\mathrm{RT}_d(\{F_1|_\rho, \ldots, F_m|_\rho\}) \geq k$. Then, there is a 1:1 mapping*

$$\theta : S \to \mathcal{R} \times [3r]^k \times \{0,1\}^k \times \{0,1\}^k \times [m]^{\lceil k/d+1 \rceil}$$

given by $\theta : \rho \mapsto (\rho\sigma, \beta, \pi, \tau, \mathcal{I})$ where σ fixes exactly k additional variables that weren't fixed by ρ.

Proof of Theorem 40, assuming Lemma 42. For a (fixed) restriction $\rho \in \mathcal{R}$ we denote by $\mathbf{Pr}[\rho]$ the probability to sample ρ when sampling a restriction from the distribution $\mathcal{R}_p$. For a (fixed) set of restrictions $A \subseteq \mathcal{R}$ we denote by $\mathbf{Pr}[A]$ the probability to sample a restriction in A when sampling a restriction from the distribution $\mathcal{R}_p$. Recall that by the definition of $\mathcal{R}_p$, we have $\mathbf{Pr}[\rho] = p^a \cdot \left(\frac{1-p}{2}\right)^{b+c}$ where a, b and c are the number of $*$'s, 0's and 1's in ρ respectively.

For a fixed value of β, π, τ, and $\mathcal{I}$, consider the set $S' = S_{\beta,\pi,\tau,\mathcal{I}} := \{\rho \in S \mid \exists \rho' : \theta(\rho) = (\rho', \beta, \pi, \tau, \mathcal{I})\}$. Since θ is 1:1 (Lemma 42), the first component $\theta_1 : \rho \mapsto \rho\sigma$ is also 1:1 on the set S'.[6] This implies that $\mathbf{Pr}[\theta_1(S')] = \sum_{\rho \in S'} \mathbf{Pr}[\theta_1(\rho)]$. By the definition of $\mathcal{R}_p$, for any $\rho \in \mathcal{R}$ and any σ that fixes k additional variables that were free in ρ, we have $\mathbf{Pr}[\rho\sigma] = \left(\frac{1-p}{2p}\right)^k \cdot \mathbf{Pr}[\rho]$. We get

$$1 \geq \mathbf{Pr}[\theta_1(S')] = \sum_{\rho \in S'} \mathbf{Pr}[\theta_1(\rho)] = \sum_{\rho \in S'} \mathbf{Pr}[\rho] \cdot \left(\frac{1-p}{2p}\right)^k = \mathbf{Pr}[S'] \cdot \left(\frac{1-p}{2p}\right)^k ,$$

hence, $\mathbf{Pr}[S'] \leq \left(\frac{2p}{1-p}\right)^k$. Taking a union bound over all possible $\beta, \pi, \tau, \mathcal{I}$ we get, as desired,

$$\mathbf{Pr}[S] \leq \sum_{\beta,\pi,\tau,\mathcal{I}} \mathbf{Pr}[S_{\beta,\pi,\tau\mathcal{I}}] \leq (3r)^k \cdot 2^k \cdot 2^k \cdot m^{\lceil k/(d+1) \rceil} \cdot \left(\frac{2p}{1-p}\right)^k . \quad \blacktriangleleft$$

Proof of Lemma 42. Let $\rho \in S$ be a restriction such that $\mathrm{RT}_d(\{F_1|_\rho, \ldots, F_m|_\rho\}) \geq k$. We describe in detail how to map ρ into $(\rho\sigma, \beta, \pi, \tau, \mathcal{I})$, where $\sigma \in \mathcal{R}, \beta \in [3r]^k, \pi \in \{0,1\}^k, \tau \in \{0,1\}^k$, and $\mathcal{I} \in [m]^{\lceil k/(d+1) \rceil}$. Then, we shall describe how to decode from $(\rho\sigma, \beta, \pi, \tau, \mathcal{I})$ the restriction ρ, showing that the mapping is 1:1.

Encoding. We are going to choose a sequence of k variables that weren't fixed by ρ, and assign them values according to three adversarial strategies:

Global Strategy. This strategy ensures that $\mathrm{RT}_d(\{F_1|_\rho, \ldots, F_m|_\rho\}) \geq k$. We will denote its answers by $\pi_1, \ldots, \pi_k \in \{0,1\}$.

Local Strategy. This will be the local adversary strategy based on one **DNF** we are focusing on. We will denote its answers by $\tau_1, \ldots, \tau_k \in \{0,1\}$.

Bread-Crumbs Strategy. The objective of this strategy is to leave the necessary traces, so that the mapping will be invertible. We will denote its answers by $\sigma_1, \ldots, \sigma_k \in \{0,1\}$.

We consider the following iterative encoding process, which is divided into phases. Each phase, except for maybe the last phase, contains at least $d+1$ steps. In each phase, t, we will focus on one specific **DNF** out of $F_1, \ldots, F_m$, and identify a sequence of variables T_t of length $d_t \geq d+1$ to be queried. The strings $\pi^t, \tau^t, \sigma^t \in \{0,1\}^{d_t}$ will be the answers to the sequence of queries T_t according to the Global, Local or Bread-Crumbs strategies, respectively.

At phase $t = 1, 2, \ldots$, we consider the restriction $\rho_t = \rho(T_1 \to \pi^1) \ldots (T_{t-1} \to \pi^{t-1})$. We identify some **DNF**, F_{i_t}, whose canonical decision tree depth under ρ_t is $d_t \geq d+1$ (if no such **DNF** exists, then we stop). We add i_t to $\mathcal{I}$. Next, we run the canonical decision tree on F_{i_t} and ρ_t, answering according to the local adversarial strategy which keeps F_{i_t} undetermined after less than d_t queries.

[6] Since a collision in θ_1 on S' implies a collision in θ.

We initialize $T_t := \emptyset$ and τ^t, σ^t to be empty-strings. In each step, we find the first term $\mathcal{T}$ in F_{i_t} which is not equivalent to 0 under $\rho_t(T_t \to \tau^t)$. By the assumption that F_{i_t} has canonical decision tree depth d_t, we get that $\mathcal{T}$ is not equivalent to 1 either. Let A be the non-empty set of variables whose literals appear in $\mathcal{T}$ and are unassigned by ρ_t. We order the set A in some canonical order. For each $x_j \in A$, let $j_{\text{ind}} \in [r]$ be the index of the literal containing x_j in term $\mathcal{T}$. We let $j_{\text{type}} = 1$ if x_j is the last variable to be queried in the DNF F_{i_t}, otherwise $j_{\text{type}} = 2$ if x_j is the last variable in A, and otherwise $j_{\text{type}} = 3$. For each $x_j \in A$, according to the A's order, we add $(j_{\text{ind}}, j_{\text{type}})$ to β. In addition, we query the local adversary according to the variables in A under the restriction $\rho_t \cdot (T_t \to \tau^t)$ and update τ^t to contain its new answers. We concatenate to σ^t the values to the variables in A that satisfy $\mathcal{T}$ (these are the "bread-crumbs"). We update $T_t := T_t \cup A$, and continue with $\rho_t \cdot (T_t \to \tau^t)$ until querying d_t variables.

After ending the phase, we ask the global adversary the sequence of queries in T_t (by the order they were asked) and consider its sequence of answers as π^t. We continue to the next phase with $\rho_{t+1} = \rho(T_1 \to \pi^1) \dots (T_t \to \pi^t)$ (i.e., we "discard" the answers to T_t according to the local adversary and add the answers according to the global adversary). We stop the encoding process after querying k variables overall, even if we are in the middle of a phase.

We show by induction that $\text{RT}_d(\{F_1|_{\rho_t}, \dots, F_m|_{\rho_t}\}) \geq k - \sum_{i=1}^{t-1} d_i$. This is trivially true for $t = 1$ since this is equivalent to the assumption that $\rho \in S$. Assuming it is true for t, we show that it is true for $t+1$. Since $\text{RT}_d(\{F_1|_{\rho_t}, \dots, F_m|_{\rho_t}\}) \geq k - \sum_{i=1}^{t-1} d_i$ it means that there exists a set of answers for T_t, namely π^t, under which $\text{RT}_d(\{F_1|_{\rho_{t+1}}, \dots, F_m|_{\rho_{t+1}}\}) \geq k - \sum_{i=1}^{t-1} d_i - |T_t| = k - \sum_{i=1}^{t} d_i$, which completes the induction.

Let p be the number of phases in the encoding process. By the above process, we get that $\pi = \pi^1 \dots \pi^p \in \{0,1\}^k$, $\tau = \tau^1 \dots \tau^p \in \{0,1\}^k$, $\beta \in [3r]^k$, $\sigma := (T_1 \to \sigma^1) \dots (T_p \to \sigma^p)$ fixes k additional variables to those fixed by ρ, and $\mathcal{I}$ is a sequence of p indices from $[m]$. In addition, $p \leq \lceil k/(d+1) \rceil$ since in each phase, except for maybe the last phase, we query at least $d+1$ variables and overall we query at most k variables. If less than $\lceil k/(d+1) \rceil$ phases exists, we may pad $\mathcal{I}$ with 1's.

Decoding. We wish to show that θ is 1:1. Let $(\rho\sigma, \beta, \pi, \tau, \mathcal{I})$ be an image of θ; we will show how to decode ρ from this image. It is enough to show by induction on $t = 1, \dots, p$, that we can recover $T_1, \dots, T_t$, since this allows to reconstruct ρ by simply setting the values of $\bigcup_{i=1}^{p} T_i$ to $*$ in $\rho\sigma$.

Assuming we already recovered $T_1, \dots, T_{t-1}$ correctly, we show how to decode T_t as well. Knowing $T_1, \dots, T_{t-1}$ allows the decoder to define $\rho'_t := \rho(T_1 \to \pi^1) \dots (T_{t-1} \to \pi^{t-1})(T_t \to \sigma^t) \dots (T_p \to \sigma^p)$ by replacing the assignment of $T_1, \dots, T_{t-1}$ in $\rho\sigma$ according to $\pi^1, \dots, \pi^t$. Using the set of indices $\mathcal{I}$, we know i_t, i.e. the index of the **DNF** out of $F_1, \dots, F_m$ that was considered by the encoding process at phase t. We show that the first term in F_{i_t} under ρ'_t which is not equivalent to 0 is the same as the first such term under ρ_t (recall that $\rho_t := \rho(T_1 \to \pi^1) \dots (T_{t-1} \to \pi^{t-1})$). Let i' be the index of the first nonzero term in F_{i_t} under ρ_t. Then, all terms prior to i' were fixed to 0 under ρ_t and this remains true when we refine ρ_t to ρ'_t. In addition, since σ^t satisfies all the literals in term i' which are unassigned by ρ_t (except if we finished the entire encoding process while in the middle of processing this term, in this case σ^t fixes some of the free variables to satisfy the literals and the rest remain free), we get that the term with index i' in F_{i_t} is not equivalent to 0 under ρ'_t. Thus, we identified the term i' correctly. We collect indices from β until reaching type 1 or 2, which yields the set of variables the encoder sets when processing term i'. We replace the assignment for these variables to be according to τ instead of according to σ.

We continue this way by identifying the next term the encoder examined, and decode the set of variables fixed in the encoding process, according to the information stored in β. This allows us to continue decoding the set T_t which completes the proof. ◀

▶ Remark. We remark that the information we are avoiding to store[7] is the index of the term on which a certain **DNF** is not fixed under a restriction ρ. We are using the Bread-Crumbs partial assignment σ to satisfy all the literals that are unassigned in this term, in order to allow the identification of the term in the decoding process. Once the term is known, we can encode/decode a variable using a number in $[r]$ rather than a number in $[n]$, which is much more "efficient" to encode. Storing the index to the DNF we are considering at each phase may seem "expensive". However, we are recording such an index at most once every $d+1$ consecutive steps, making this reasonable.

Acknowledgements. I wish to thank my advisor Ran Raz for many helpful discussions, for his encouragement and his support. I thank Johan Håstad and Roei Tell for helpful discussions. I thank the anonymous referees for helpful comments.

References

1 S. Aaronson. BQP and the polynomial hierarchy. In *STOC*, pages 141–150, 2010.
2 M. Ajtai. Σ_1^1-formulae on finite structures. *Annals of Pure and Applied Logic*, 24:1–48, 1983.
3 N. Alon, O. Goldreich, J. Håstad, and R. Peralta. Simple construction of almost k-wise independent random variables. *Random Structures and Algorithms*, 3(3):289–304, 1992.
4 L. M. J. Bazzi. Polylogarithmic independence can fool DNF formulas. *SIAM J. Comput.*, 38(6):2220–2272, 2009.
5 P. Beame. A switching lemma primer, 1994.
6 R. B. Boppana. The average sensitivity of bounded-depth circuits. *Inf. Process. Lett.*, 63(5):257–261, 1997.
7 M. Braverman. Polylogarithmic independence fools AC^0 circuits. *J. ACM*, 57(5):28:1–28:10, 2010.
8 G. Cohen, A. Ganor, and R. Raz. Two sides of the coin problem. In *APPROX-RANDOM*, pages 618–629, 2014.
9 A. De, O. Etesami, L. Trevisan, and M. Tulsiani. Improved pseudorandom generators for depth 2 circuits. In *APPROX-RANDOM*, pages 504–517, 2010.
10 Y. Filmus. Smolensky's lower bound. Unpublished Manuscript, 2010.
11 M. Furst, J. B. Saxe, and M. Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17(1):13–27, apr 1984.
12 O. Goldreich and L. A. Levin. A hardcore predicate for all one-way functions. In *STOC*, pages 25–32, 1989.
13 P. Harsha and S. Srinivasan. On polynomial approximations to AC^0. In *RANDOM 2016*, pages 32:1–32:14, 2016.
14 J. Håstad. Almost optimal lower bounds for small depth circuits. In *STOC*, pages 6–20, 1986. doi:10.1145/12130.12132.
15 J. Håstad. A slight sharpening of LMN. *J. Comput. Syst. Sci.*, 63(3):498–508, 2001. doi:10.1006/jcss.2001.1803.
16 J. Håstad. On the correlation of parity and small-depth circuits. *SIAM J. Comput.*, 43(5):1699–1708, 2014.

[7] And we have good reasons to do so, since this will result in a non-effective switching lemma.

17 R. Impagliazzo and V. Kabanets. Fourier concentration from shrinkage. In *CCC*, pages 321–332, 2014.

18 R. Impagliazzo, W. Matthews, and R. Paturi. A satisfiability algorithm for AC^0. In *SODA*, pages 961–972, 2012.

19 R. Kaas and J. M. Buhrman. Mean, median and mode in binomial distributions. *Statistica Neerlandica*, 34(1):13–18, 1980.

20 J. Kahn, G. Kalai, and N. Linial. The influence of variables on Boolean functions. In *FOCS*, pages 68–80, 1988.

21 E. Kushilevitz and Y. Mansour. Learning decision trees using the Fourier spectrum. *SIAM J. Comput.*, 22(6):1331–1348, 1993.

22 N. Linial, Y. Mansour, and N. Nisan. Constant depth circuits, Fourier transform and learnability. *J. ACM*, 40(3):607–620, 1993.

23 M. Luby and B. Velickovic. On deterministic approximation of DNF. *Algorithmica*, 16(4/5):415–433, 1996.

24 O. Lupanov. Implementing the algebra of logic functions in terms of constant depth formulas in the basis $\{\&, \vee, \neg\}$. *Dokl. Akad. Nauk. SSSR*, 136:1041–1042, 1961. In Russian.

25 Y. Mansour. An $O(n^{\log\log n})$ learning algorithm for DNF under the uniform distribution. *J. Comput. Syst. Sci.*, 50(3):543–550, 1995. doi:10.1006/jcss.1995.1043.

26 J. Naor and M. Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM J. on Computing*, 22(4):838–856, 1993.

27 R. O'Donnell. *Analysis of boolean functions*. Cambridge University Press, 2014.

28 R. O'Donnell and K. Wimmer. Approximation by DNF: examples and counterexamples. In *ICALP*, pages 195–206, 2007.

29 A. A. Razborov. Bounded arithmetic and lower bounds in boolean complexity. In *Feasible Mathematics II*, volume 13 of *Progress in Computer Science and Applied Logic*, pages 344–386. Birkhäuser Boston, 1995.

30 A. A. Razborov. A simple proof of Bazzi's theorem. *TOCT*, 1(1), 2009.

31 R. Shaltiel and E. Viola. Hardness amplification proofs require majority. *SIAM J. Comput.*, 39(7):3122–3154, 2010.

32 R. Smolensky. On representations by low-degree polynomials. In *FOCS 1993*, pages 130–138, 1993.

33 A. Tal. Shrinkage of de Morgan formulae from quantum query complexity. In *FOCS*, pages 551–560, 2014.

34 N. Thapen. Notes on switching lemmas. Unpublished Manuscript, 2009. URL: http://users.math.cas.cz/~thapen/switching.pdf.

35 L. Trevisan and T. Xue. A derandomized switching lemma and an improved derandomization of AC0. In *CCC*, pages 242–247, 2013.

36 A. C. Yao. Separating the polynomial hierarchy by oracles. In *FOCS*, pages 1–10, 1985.

A Equivalent Expressions for the T-th Discrete Derivatives

▶ **Claim** (Claim 25, restated)**.**

$$D_T f(x) = \frac{1}{2^{|T|}} \sum_{z\in\{-1,1\}^T} f(x^{(T\mapsto z)}) \cdot \prod_{i\in T} z_i = \sum_{S\supseteq T} \hat{f}(S) \cdot \prod_{i\in S\setminus T} x_i$$

where $x^{(T\mapsto z)}$ is the vector in $\{-1,1\}^n$ whose i-th coordinate equals z_i whenever $i \in T$, and x_i otherwise.

Proof. We prove by induction on the size of T. For $T = \emptyset$ the claim trivially holds. For $T = \{j_1, \ldots, j_k\}$, let $T' = \{j_2, \ldots, j_k\}$ and $g = D_{T'} f$, then $D_T f = D_{j_1} D_{T'} f = D_{j_1} g$. By the definition of the j_1-th derivative, we have

$$D_T f(x) = \frac{g(x^{(j_1 \mapsto 1)}) - g(x^{(j_1 \mapsto -1)})}{2} .$$

By the induction hypothesis, this equals

$$\begin{aligned} D_T f(x) &= \frac{1}{2} \cdot \left(D_{T'} f(x^{(j_1 \mapsto 1)}) - D_{T'} f(x^{(j_1 \mapsto -1)}) \right) \\ &= \frac{1}{2} \frac{1}{2^{k-1}} \cdot \left(\begin{array}{l} \sum_{z' \in \{-1,1\}^{T'}} f\left(\left(x^{(j_1 \mapsto 1)} \right)^{(T' \mapsto z')} \right) \prod_{i \in T'} z'_i \\ - \sum_{z' \in \{-1,1\}^{T'}} f\left(\left(x^{(j_1 \mapsto -1)} \right)^{(T' \mapsto z')} \right) \prod_{i \in T'} z'_i \end{array} \right) \\ &= \frac{1}{2^k} \sum_{z \in \{-1,1\}^T} f(x^{(T \mapsto z)}) \prod_{i \in T} z_i . \end{aligned}$$

As for the second item, by induction, $g(x) = \sum_{S \supseteq T'} \hat{f}(S) \cdot \prod_{i \in S \setminus T'} x_i$. Thus,

$$\begin{aligned} D_T f(x) = \frac{g(x^{(j_1 \mapsto 1)}) - g(x^{(j_1 \mapsto -1)})}{2} &= \frac{1}{2} \sum_{S \supseteq T'} \hat{f}(S) \cdot \prod_{i \in S \setminus T} x_i \cdot \begin{cases} 1 - (-1), & j_1 \in T' \\ 1 - 1, & otherwise \end{cases} \\ &= \sum_{S \supseteq T} \hat{f}(S) \cdot \prod_{i \in S \setminus T} x_i . \end{aligned}$$

◀

B Rephrasing Braverman's Result

▶ **Lemma 43** ([7, Lemma 8]). *Let ν be any probability distribution on $\{0,1\}^n$. For a circuit of depth d and size m computing a function F, for any s, there is a degree $r = (s \cdot \log(m))^d$ polynomial f and a Boolean function $\mathcal{E}_\nu$ computable by a circuit of depth $\leq d + 3$ and size $O(m^2 r)$ such that*

1. $\boldsymbol{Pr}_\nu[\mathcal{E}_\nu(x) = 1] < 0.82^s \cdot m$, *and*
2. *whenever $\mathcal{E}_\nu = 0$, $f(x) = F(x)$.*

▶ **Proposition 44** ([7, Prop. 9]). *In Lemma 43, for $s \geq \log(m)$, $\|f\|_\infty < (2m)^{\deg(f)-2} = (2m)^{(s \log(m))^d - 2}$*

▶ **Lemma 45** ([7, Rephrasing of Lemma 10]). *Let F be computed by a circuit of depth d and size m. Let s_1, s_2 be two parameters with $s_1 \geq \log(m)$. Let μ be any probability distribution on $\{0,1\}^n$, and $U_{\{0,1\}^n}$ be the uniform distribution on $\{0,1\}^n$. Set*

$$\nu := \frac{1}{2} \left(\mu + U_{\{0,1\}^n} \right) .$$

Let $\mathcal{E}_\nu$ be the function from Lemma 8 with $s = s_1$. Set $F' = F \vee \mathcal{E}_\nu$. Then, there is a polynomial f' of degree $r_f = (s_1 \cdot \log m)^d + s_2$, such that

1. $\boldsymbol{Pr}_\mu[F \neq F'] < 2 \cdot 0.82^{s_1} \cdot m$
2. $\boldsymbol{Pr}_U[F \neq F'] < 2 \cdot 0.82^{s_1} \cdot m$
3. $\|F' - f'\|_2^2 \leq 0.82^{s_1} \cdot (4m) + 2^{4(s_1 \cdot \log m)^d \log m} \cdot \mathrm{tail}(m^3, d+3, s_2)$, *and*
4. $f'(x) = 0$ *whenever* $F'(x) = 0$.

Proof. The first two properties follow from Lemma 43 directly, since

$$\mathop{\mathbf{Pr}}_{\mu}[\mathcal{E}_\nu = 1], \mathop{\mathbf{Pr}}_{U_n}[\mathcal{E}_\nu = 1] \leq 2 \cdot \mathop{\mathbf{Pr}}_{\nu}[\mathcal{E}_\nu = 1] \leq 2 \cdot 0.82^{s_1} m \ .$$

Let f be the degree $(s_1 \cdot \log m))^d$ approximation of F from Lemma 43. By Proposition 44,

$$\|f\|_\infty < (2m)^{(s_1 \cdot \log m)^d - 2} < 2^{2(s_1 \log m)^d \log(m) - 2} \ .$$

Let $\tilde{\mathcal{E}}_\nu$ be the truncated Fourier expansion of $\mathcal{E}_\nu$ of degree s_2. We have

$$\|\mathcal{E}_\nu - \tilde{\mathcal{E}}_\nu\|_2^2 \leq \mathrm{tail}(m^3, d+3, s_2) \ .$$

Let

$$f' := f \cdot (1 - \tilde{\mathcal{E}}_\nu)$$

Then $f' = 0$ whenever $F' = 0$ (since $(F' = 0) \implies (\mathcal{E}_\nu = 0, F = 0) \implies (f = 0) \implies (f' = 0)$). It remains to estimate $\|F' - f'\|_2^2$:

$$\begin{aligned}
\|F' - f'\|_2^2 &\leq 2 \cdot \|F' - f \cdot (1 - \mathcal{E}_\nu)\|_2^2 + 2 \cdot \|f \cdot (1 - \mathcal{E}_\nu) - f'\|_2^2 \\
&= 2 \cdot \|\mathcal{E}_\nu\|_2^2 + 2 \cdot \|f \cdot (\mathcal{E}_\nu - \tilde{\mathcal{E}}_\nu)\|_2^2 \\
&\leq 2 \cdot \mathbf{Pr}[\mathcal{E}_\nu = 1] + 2 \cdot \|f\|_\infty^2 \cdot \|\mathcal{E}_\nu - \tilde{\mathcal{E}}_\nu\|_2^2 \\
&\leq 0.82^{s_1} \cdot (4m) + 2^{4(s_1 \cdot \log m)^d \log m} \cdot \mathrm{tail}(m^3, d+3, s_2),
\end{aligned}$$

which completes the proof. ◀

▶ **Theorem 46** ([7, Rephrasing of Main Theorem]). *Let $s_1, s_2 \geq \log m$ be any parameters. Let F be a Boolean function computed by a circuit of depth d and size m. Let μ be an r-independent distribution where*

$$r = r(s_1, s_2, d) = 2((s_1 \cdot \log m)^d + s_2)$$

then

$$|\mathop{\boldsymbol{E}}_{\mu}[F] - E[F]| \leq \varepsilon(s_1, s_2, d),$$

where $\varepsilon(s_1, s_2, d) = 0.82^{s_1} \cdot (6m) + 2^{4(s_1 \cdot \log m)^d \log m} \cdot \mathrm{tail}(m^3, d+3, s_2)$

Proof of Theorem 46. Denote by $\varepsilon_1 := 0.82^{s_1} \cdot (2m)$ and

$$\varepsilon_2 := 0.82^{s_1} \cdot (4m) + 2^{4(s_1 \cdot \log m)^d \log m} \cdot \mathrm{tail}(m^3, d+3, s_2) \ .$$

Applying Lemma 45 with parameters s_1 and s_2 gives

$$\|F' - f'\|_2^2 \leq \varepsilon_2 \ .$$

Now take $f'_\ell := 1 - (1 - f')^2$. Then $f'_\ell \leq 1$ and $f'_\ell = 0$ whenever $F' = 0$, hence $f'_\ell \leq F'$. To estimate $\mathbf{E}[F'(x) - f'_\ell(x)]$ we note that $F'(x) - f'_\ell(x)$ equals 0 whenever $F' = 0$, and is equal to

$$F'(x) - f'_\ell(x) = (1 - f'(x))^2 = (F'(x) - f'(x))^2$$

whenever $F' = 1$. We get

$$\mathbf{E}[F'(x) - f'_\ell(x)] \leq \|F' - f'\|_2^2 \leq \varepsilon_2 \ .$$

In addition, $\deg(f'_\ell(x)) \leq 2(s_2 + (s_1 \cdot \log m)^d)$.

To finish the proof, if μ is a $\left(2 \cdot (s_2 + (s_1 \cdot \log m)^d)\right)$-wise independent distribution then

$$\begin{aligned}\mathop{\mathbf{E}}_{\mu}[F(x)] &\geq \mathop{\mathbf{E}}_{\mu}[F'(x)] - \varepsilon_1 \geq \mathop{\mathbf{E}}_{\mu}[f'_\ell(x)] - \varepsilon_1 =^* \mathbf{E}[f'_\ell(x)] - \varepsilon_1 \\ &= \mathbf{E}[F'(x)] - \mathbf{E}[F'(x) - f'_\ell(x)] - \varepsilon_1 \geq \mathbf{E}[F'(x)] - \varepsilon_2 - \varepsilon_1 \geq \mathbf{E}[F(x)] - \varepsilon_2 - \varepsilon_1\end{aligned}$$

where we used in * the fact that $\deg(f'_\ell) \leq 2(s_2 + (s_1 \cdot \log m)^d)$ and μ is $\deg(f'_\ell)$-wise independent. In a similar way, one can show $\mathbf{E}_\mu[F(x)] \leq \mathbf{E}[F(x)] + \varepsilon_1 + \varepsilon_2$. Combining both cases we get

$$|\mathop{\mathbf{E}}_{\mu}[F] - E[F]| \leq \varepsilon_1 + \varepsilon_2 = \varepsilon(s_1, s_2, d) \ . \qquad \blacktriangleleft$$

C Improving the Analysis of De, Etesami, Trevisan and Tulsiani

De et al. [9] proved that any ε-biased distribution δ-fools depth-2 circuits (**DNFs** or **CNFs**) of size m, for some $\varepsilon = \varepsilon(\delta, m)$. In fact, their work shows that generators of ε-biased distributions are the best known pseudorandom generators fooling depth-2 circuits. We are able to improve their analysis slightly, getting an optimal dependence between ε and δ.

Some notation is needed first. Throughout this section (and the next), we shall think of Boolean functions as functions $f : \{0,1\}^n \to \mathbb{R}$ (as opposed to $f : \{-1,1\}^n \to \mathbb{R}$). We can identify each function $f : \{0,1\}^n \to \mathbb{R}$ with a function $\tilde{f} : \{-1,1\}^n \to \mathbb{R}$ by $\tilde{f}(y_1, \ldots, y_n) = f(\frac{1-y_1}{2}, \ldots, \frac{1-y_n}{2})$ or equivalently $f(x_1, \ldots, x_n) = \tilde{f}((-1)^{x_1}, \ldots, (-1)^{x_n})$. When talking about the Fourier expansion of f, we mean the Fourier expansion of $\tilde{f}$ as defined in Section 2. In this notation, $f(x) = \sum_{S \subseteq [n]} \hat{f}(S) \cdot (-1)^{\sum_{i \in S} x_i}$.

Next, we discuss **DNFs** and **CNFs**. Disjunctive normal forms (**DNFs**) are expressions of the form $F(x) = \bigvee_{i=1}^m t_i(x)$ where each term $t_i(x)$ is an AND of some literals from $x_1, \ldots, x_n, \neg x_1, \ldots, \neg x_n$. If any term in F is an AND of at most w literals, then we say that F is of width w, and we call F a w-**DNF**. Similarly conjunctive normal forms (**CNFs**) are expressions of the form $F(x) = \bigwedge_{i=1}^m c_i(x)$ where each clause c_i is an OR of some literals. We define w-**CNFs** similarly to w-**DNFs**. The size of a **DNF** (**CNF**, resp.) is the number of terms (clauses, resp.) in it, i.e., m in the examples above.

Recall the definition of the spectral norm of a Boolean function $L_1(f) = \sum_S |\hat{f}(S)|$ and denote by $L_1^*(f) = \sum_{S \neq \emptyset} |\hat{f}(S)|$. We denote by U_n the uniform distribution over $\{0,1\}^n$.

We cite a proposition and two lemmata from the work of De et al. [9].

▶ **Proposition 47** ([9, Prop. 2.6]). *Suppose $f, f_\ell, f_u : \{0,1\}^n \to \mathbb{R}$ are three functions such that for every $x \in \{0,1\}^n$ we have $f_\ell(x) \leq f(x) \leq f_u(x)$. Furthermore, assume $\boldsymbol{E}_{x \sim U_n}[f(x) - f_\ell(x)] \leq \delta$ and $\boldsymbol{E}_{x \sim U_n}[f_u(x) - f(x)] \leq \delta$. Let $l = \max(L_1^*(f_\ell), L_1^*(f_u))$. Then, any ε-biased probability distribution $(\delta + \varepsilon l)$-fools f.*

▶ **Lemma 48** ([9, Lemma 4.3]). *Let $f : \{0,1\}^n \to \{0,1\}$ be a* **DNF** *with m terms and $g : \{0,1\}^n \to \mathbb{R}$ be such that: $L_1(g) \leq l_1$, $\|f - g\|_2^2 \leq \varepsilon_1$ and $g(x) = 0$ whenever $f(x) = 0$. Then, we can get $f_\ell, f_u : \{0,1\}^n \to \mathbb{R}$ such that*

- $\forall x, f_\ell(x) \leq f(x) \leq f_u(x)$
- $\boldsymbol{E}_{x \sim U_n}[f_u(x) - f(x)] \leq m \cdot \varepsilon_1$ *and* $\boldsymbol{E}_{x \sim U_n}[f(x) - f_\ell(x)] \leq m \cdot \varepsilon_1$.
- $L_1(f_\ell), L_1(f_u) \leq (m+1)(l_1+1)^2 + 1$.

▶ **Lemma 49** ([9, Lemma 4.4]). *Let $f : \{0,1\}^n \to \{0,1\}$ be a* **DNF** *with m terms and width-w. Suppose for every* **DNF** *with at most m terms and width-w, f_1, there is a function $g_1 : \{0,1\}^n \to \mathbb{R}$ such that: $L_1(g_1) \leq l_2$ and $\|f_1 - g_1\|_2^2 \leq \varepsilon_2$. Then, we can get $g : \{0,1\}^n \to \mathbb{R}$ such that $L_1(g) \leq m \cdot (l_2 + 1)$, $\|f - g\|_2^2 \leq m^2 \cdot \varepsilon_2$ and $g(x) = 0$ whenever $f(x) = 0$.*

De et al. [9] used Lemma 49 with a bound on the width of the approximated **DNF**. We will use Lemma 49 without any assumption on the width.

The following is a corollary of Thm. 37.

▶ **Corollary 50.** *Let $f : \{0,1\}^n \to \{0,1\}$ be a* **DNF** *of size m and $\varepsilon_2 > 0$. Then, there is a function $g_1 : \{0,1\}^n \to \mathbb{R}$ such that $\boldsymbol{E}[(f - g_1)^2] \leq \varepsilon_2$ and $L_1(g_1) = 2^{O(\log m \cdot \log\log m \cdot \log(1/\varepsilon_2))}$.*

Proof. According to Thm. 37, there is a set $\mathcal{F}$ of coefficients such that $|\mathcal{F}| \leq 2^{O(\log m \cdot \log\log m \cdot \log(1/\varepsilon_2))}$ and $\sum_{S \notin \mathcal{F}} \hat{f}(S)^2 \leq \varepsilon_2$. Hence, $g_1(x) = \sum_{S \in \mathcal{F}} \hat{f}(S) \cdot (-1)^{\sum_{i \in S} x_i}$ is an approximation of f with

$$\mathop{\mathbf{E}}_{x}[(f(x) - g_1(x))^2] = \sum_{S \notin \mathcal{F}} \hat{f}(S)^2 \leq \varepsilon_2 \ .$$

where we used Parseval's identity. Since each Fourier coefficient is at most 1 in absolute value $L_1(g_1) = \sum_{S \in \mathcal{F}} |\hat{f}(S)| \leq |\mathcal{F}|$, which completes the proof. ◀

The following theorem is our refinement of [9, Thm. 4.1].

▶ **Theorem 51.** *Let f be a* **DNF** *formula with m terms. Then, f is δ-fooled by any ε-biased distribution where $\varepsilon = 2^{O(\log m \cdot \log(m/\delta) \cdot \log\log m)}$.*

Proof. Set $\varepsilon_2 = \delta/2m^3$ and $\varepsilon_1 = \delta/2m$. By applying Corollary 50 for every **DNF** formula of size m, f_1, there exists a function $g_1 : \{0,1\}^n \to \mathbb{R}$ such that

- $\mathbf{E}[(f_1 - g_1)^2] \leq \varepsilon_2$
- $L_1(g_1) \leq 2^{O(\log m \cdot \log\log m \cdot \log(1/\varepsilon_2))} = 2^{O(\log m \cdot \log\log m \cdot \log(m/\delta))}$

We apply Lemma 49 with width n (this is a trivial choice of width, since all **DNFs** on n variables are of width at most n without loss of generality), $\varepsilon_2 = \delta/2m^3$ and $l_2 = 2^{O(\log m \cdot \log\log m \cdot \log(m/\delta))}$. Then, we get the existence of a function $g : \{0,1\}^n \to \mathbb{R}$ such that $g(x) = 0$ whenever $f(x) = 0$ and $\mathbf{E}[(g - f)^2] \leq m^2\varepsilon_2 = \delta/2m$. and $L_1(g) \leq (l_2+1) \cdot m = 2^{O(\log m \cdot \log\log m \cdot \log(m/\delta))}$. Then, we apply Lemma 48 with g, $\varepsilon_1 = \delta/2m$ and $l_1 = L_1(g) = 2^{O(\log m \cdot \log\log m \cdot \log(m/\delta))}$ to get a sandwiching approximation of f by f_ℓ and f_u such that

- $\forall x : f_\ell(x) \leq f(x) \leq f_u(x)$
- $\mathbf{E}_{x \sim U_n}[f_u(x) - f(x)] \leq m \cdot \varepsilon_1 = \delta/2$ and $\mathbf{E}_{x \sim U_n}[f(x) - f_\ell(x)] \leq m \cdot \varepsilon_1 = \delta/2$.
- $L_1(f_u), L_1(f_\ell) \leq (l_1 + 1)^2 \cdot (m + 1) + 1 = 2^{O(\log m \cdot \log\log m \cdot \log(m/\delta))}$.

Denote by $l = (l_1 + 1) \cdot (m + 1) + 1$. Applying Prop. 47, we get that any $\varepsilon = \delta/(2l) = 2^{-O(\log m \cdot \log\log m \cdot \log(m/\delta))}$ biased distribution γ-fools f, where $\gamma = \delta/2 + \varepsilon \cdot l \leq \delta$. ◀

It is well-known from the works of [26, 3] that ε-biased distributions on n bits may be sampled using a $O(\log n + \log(1/\varepsilon))$-seed length, which gives the following corollary.

▶ **Theorem 52.** *There exists a polynomial time pseudorandom generator $\mathcal{G}$ of seed length $O(\log n + \log m \cdot \log\log m \cdot \log(m/\delta))$ that δ-fools all* **DNFs** *of size m on n variables.*

Note that by de Morgan laws every **DNF** of size m is the negation of a **CNF** of size m, and vice versa. Hence, any pseudorandom generator that fools **DNFs** also fools **CNFs**.

D Improving the Generator of Trevisan and Xue

In this section, we revisit the pseudorandom generator of Trevisan and Xue [35] that ε-fools $\mathbf{AC^0}$ circuits of size M and depth d. We improve its seed-length from $O(\log^{d+3}(M/\varepsilon) \cdot \log(n/\varepsilon))$ to $O(\log^{d+1}(M/\varepsilon) \cdot \log n)$ by two observations in addition to the improved analysis of the generator of De et al. (see Thm. 52).

We start by explaining the sampling process of Trevisan and Xue's generator at a high-level. The generator applies $O(\log^{d-1}(M/\varepsilon) \cdot \log(n/\varepsilon))$ pseudorandom restrictions iteratively, where each pseudorandom restriction fixes each variable (that wasn't already fixed) with probability $\Theta(1/\log^{d-1}(M/\varepsilon))$. The seed length required per step is $\tilde{O}(\log^4(M/\varepsilon))$. Each pseudo-random restriction consists of a pseudorandom process that selects which variables to fix, in addition to a pseudorandom process that selects the values for these variables. The heart of Trevisan and Xue's analysis is a proof that the selection of which variables to fix can be done by sampling recursively d times (one per depth) from any distribution that fools **CNFs** with appropriate parameters. This is done by proving that any distribution that fools **CNFs**, also fools Håstad's switching lemma [14] (see Lemma 53 below).

Our improvement from seed-length $\tilde{O}(\log^{d+3}(M/\varepsilon) \cdot \log(n/\varepsilon))$ to $\tilde{O}(\log^{d+1}(M/\varepsilon) \cdot \log n)$ is a combination of three improvements:

- We get a factor of $\log(M/\varepsilon)$ improvement via a better analysis of the pseudorandom generator of De et al. [9] (see Section C). We get a better dependency on the error parameter ε_0 in Thm. 52, compared to the corresponding theorem of [9]. Since Trevisan and Xue use Thm. 52 with error parameter $\varepsilon_0 = 1/2^{\Theta(\log^2(M/\varepsilon))}$ that is much smaller than any polynomial in ε/M, this improvement is effective.
- We get a factor of $\log(M/\varepsilon)$ improvement by applying the switching lemma for one less step. We show that with high probability the circuit collapses to a depth-2 circuit instead of collapsing to a bounded depth decision tree. Since we are able to fool depth-2 circuits by Thm. 52, this is enough.[8]
- We replace a factor of $\log(n/\varepsilon)$ by a factor of $\log(n)$ by noting that one can continue restricting variables until less than $O(\log(1/\varepsilon))$ variables are alive, and then fix the remaining variables using a $O(\log(1/\varepsilon))$-wise independent distribution. In the original analysis, one waited until all variables were fixed.

In the rest of the section, we will use the following notation (as suggested by Trevisan and Xue [35]). A restriction may be defined (not uniquely) by two binary strings of length n: $\theta \in \{*, \square\}^n$ and $\beta \in \{0,1\}^n$, where for $i \in [n]$,

$$\rho(i) = \begin{cases} \beta(i), & \theta(i) = \square \\ *, & \theta(i) = * \end{cases}.$$

We shall identify a string $w \in \{0,1\}^{n(q+1)}$ with a restriction as follows. We partition w to (l, r) where l consists of the first qn bits of w, and r consists of the last n bits of w. We further partition $l \in \{0,1\}^{nq}$ to n blocks of q consecutive bits each. For block $i \in [n]$, we take $\theta(i) = *$ iff all the q bits in the block equal 1. We take $\beta = r$ and yield the restriction defined by (θ, β).

[8] To get another $\tilde{O}(\log(M/\varepsilon))$ improvement it is enough to construct PRGs for depth-3 circuits with seed-length $\tilde{O}(\log^3(M/\varepsilon))$. Then, one can stop one step sooner (i.e. when reaching depth-3) and apply the PRG for depth-3 on the remaining circuit.

If D is a distribution over $\{0,1\}^{(q+1)n}$, then $(\theta,\beta) \sim D$ means that we sample $w \sim D$ as a string of length $(q+1)n$ and use the aforementioned identification to get $\theta \in \{*,\square\}^n$ and $\beta \in \{0,1\}^n$. Note that sampling $w \in \{0,1\}^{n\cdot(q+1)}$ uniformly at random yields a restriction $\rho = (\theta,\beta)$ distributed according to $\mathcal{R}_p$ for $p = 2^{-q}$.

▶ **Lemma 53** ([35, Lemma 7]). *Let F be a* **CNF** *of size M and width t over n variables, $p_0 = 2^{-q_0}$ where $q_0 \in \mathbb{N}$, and D be a distribution over $\{0,1\}^{(q_0+1)n}$ that ε_0-fools all* **CNFs** *of size at most $M \cdot 2^{t\cdot(q_0+1)}$. Then,*

$$\Pr_{(\theta,\beta)\sim D}[\mathrm{DT}(F|_{\theta,\beta}) > s] \leq 2^{s+t+1} \cdot (5p_0 t)^s + \varepsilon_0 \cdot 2^{(s+1)\cdot(2t+\log M)},$$

where $\mathrm{DT}(f)$ denotes the depth of the smallest decision tree computing a function f.[9]

The following is a slight generalization of [35, Fact 9].

▶ **Fact 54** ([35, Fact 9]). *Let D_1 be a distribution over $\{0,1\}^{n_1}$ that ε_1-fools* **CNFs** *of size m on n_1 variables. Let D_2 be a distribution over $\{0,1\}^{n_2}$ that ε_2-fools* **CNFs** *of size m on n_2 variables. Let $D_1 \otimes D_2$ be the distribution over $\{0,1\}^{n_1+n_2}$ sampled by concatenating independent samples from D_1 and D_2. Then, $D_1 \otimes D_2$ is a distribution that $(\varepsilon_1+\varepsilon_2)$-fools* **CNFs** *of size m on n_1+n_2 variables.*

Proof. Let $F(X,Y)$ be a **CNF** of size m on n_1+n_2 variables, where X consists of the first n_1 variables and Y consists of the last n_2 variables. We have

$$\mathop{\mathbf{E}}_{x\sim U_{n_1}, y\sim U_{n_2}}[F(x,y)] = \mathop{\mathbf{E}}_{x\sim U_{n_1}}\Big[\mathop{\mathbf{E}}_{y\sim U_{n_2}}[F_x(y)]\Big]$$

where $F_x(\cdot)$ is the **CNF** F when the variables X are fixed to x. Note that $F_x(\cdot)$ is in itself a **CNF** of size at most m on n_2 variables. By assumption, for all values of x,

$$\mathop{\mathbf{E}}_{y\sim U_{n_2}}[F_x(y)] = \mathop{\mathbf{E}}_{y\sim D_2}[F_x(y)] \pm \varepsilon_2\ . \tag{9}$$

Similarly for any fixed assignment $Y = y$, we have

$$\mathop{\mathbf{E}}_{x\sim U_{n_1}}[F(x,y)] = \mathop{\mathbf{E}}_{x\sim D_1}[F(x,y)] \pm \varepsilon_1\ . \tag{10}$$

Combining Eqs. (9) and (10) gives

$$\begin{aligned}
\mathop{\mathbf{E}}_{x\sim U_{n_1}, y\sim U_{n_2}}[F(x,y)] &= \mathop{\mathbf{E}}_{x\sim U_{n_1}}\left[\mathop{\mathbf{E}}_{y\sim D_2}[F_x(y)] \pm \varepsilon_2\right] \\
&= \mathop{\mathbf{E}}_{y\sim D_2}\Big[\mathop{\mathbf{E}}_{x\sim U_{n_1}}[F(x,y)] \pm \varepsilon_2\Big] \\
&= \mathop{\mathbf{E}}_{y\sim D_2}\Big[\mathop{\mathbf{E}}_{x\sim D_1}[F(x,y)] \pm \varepsilon_1 \pm \varepsilon_2\Big] \\
&= \mathop{\mathbf{E}}_{x\sim D_1, y\sim D_2}[F(x,y)] \pm (\varepsilon_1+\varepsilon_2)\ .
\end{aligned}$$

◀

By induction, Fact 54 implies the following corollary.

[9] Actually, Trevisan and Xue show the stronger result where $\mathrm{DT}(f)$ is replaced by the depth of the canonical decision tree for f (see Section 7 for its definition). However, we do not benefit from this strengthening.

▶ **Corollary 55.** *Let D be a distribution over $\{0,1\}^n$ that ε-fools* **CNFs** *of size m. Let $t \in N$, and $D^{\otimes t}$ be the distribution over $\{0,1\}^{n \cdot t}$ sampled by concatenating t independent samples from D. Then, $D^{\otimes t}$ is a distribution that $(\varepsilon \cdot t)$-fools* **CNFs** *of size m on $n \cdot t$ variables.*

In the following theorems we shall assume that the circuit size M is larger than the length of the input n.

▶ **Theorem 56** ([35, Thm. 11, "Derandomized Switching Lemma for $\mathbf{AC^0}$", restated])**.** *Let C be circuit on n variables with size M, depth d and a top* OR*-gate. Let $p = 2^{-q}$, where $q \in \mathbb{N}$, and $s \in \mathbb{N}$ be some positive parameter. Assume that there exists a pseudorandom generator $\mathcal{G}$ with seed length r that ε_0-fools* **CNFs** *of size $M \cdot 2^s \cdot 2^{s \cdot (q+1)}$. Then, there exists a pseudorandom selection generator $\mathcal{G}_0$ of seed length $(d-1) \cdot r$ such that:*

- $\boldsymbol{Pr}_{\theta \sim \mathcal{G}_0, \beta \sim \mathcal{U}}[F|_{\theta,\beta}$ *is not an* s-**DNF** *of size* $\leq M \cdot 2^s]$
  $\leq M \cdot \left(2^{2s+1} \cdot \max\{(5ps)^s, (5/64)^s\} + \varepsilon_0 \cdot 2^{(s+1)\cdot(3s+\log M)}\right)$.
- *For each set of variables $T \subseteq [n]$, the probability that all variables in T are fixed is at most $(1 - p^{d-2}/64)^{|T|} + \varepsilon_0 \cdot (d-1)$.*

Proof. We shall start by adding a dummy layer next to the inputs that transforms the circuit C into a circuit C' of size at most $M \cdot n$, depth $d+1$ and bottom fan-in 1. We construct $\mathcal{G}_0$ by running $(d-1)$ iterative pseudorandom selections, using the generator of [9] in each iteration. By Fact 54, the pair (θ, β) obtained by sampling $\theta \sim \mathcal{G}$ and $\beta \in \{0,1\}^n$ uniformly at random, ε_0-fools **CNFs** of size $M \cdot 2^s \cdot 2^{s \cdot (q+1)}$. We denote by $M_1, \ldots, M_d$ the number of gates in the original circuit C at distance $1, \ldots, d$ from the inputs, respectively.

The first iteration. For the first iteration of Lemma 53, we pick $p_0 = 1/64$ and $t = 1$. The probability that under the pseudorandom restriction, one of the gates at distance 2 from the inputs cannot be computed by a decision tree of depth s is at most

$$M_1 \cdot \left(2^{s+1+1} \cdot (5/64)^s + \varepsilon_0 \cdot 2^{(s+1)(2+\log M)}\right) .$$

In the complement event, we may express each gate at distance 2 from the inputs both as an s-**DNF** and as an s-**CNF**, so we can collapse this layer with the layer above it. This simplification yields a circuit of depth d, fan-in s and does not introduce new gates at distance 2 or more from the inputs. The number of gates at distance 1 from the inputs is at most $M \cdot 2^s$, since each depth-s decision tree is an s-**DNF** of size at most 2^s (and similarly an s-**CNF** of size at most 2^s).

The other $d-2$ iterations. At iteration $i = 2, \ldots, d-1$, we apply Lemma 53 with $t = s$ and $p_0 = p$. We get that under the pseudorandom restriction, the probability that there exists a gate at distance 2 from the inputs that cannot be computed by a decision tree of depth s is at most

$$M_i \cdot \left(2^{s+s+1} \cdot (5ps)^s + \varepsilon_0 \cdot 2^{(s+1)(2s+\log(M \cdot 2^s))}\right) .$$

We are using the fact that each gate at distance 2 from the inputs computes a **CNF**/**DNF** of size at most $M \cdot 2^s$ and bottom fan-in s, an invariant that is preserved during the iterative process. Again, if a gate is computed by a decision tree of depth s then it is also computed by an s-**CNF** and by an s-**DNF** of size at most 2^s, and we may collapse the layers at distances 2 and 3 from the inputs.

Overall, after $(d-1)$-iterations, with probability at least

$$1 - M \cdot \left(2^{s+s+1} \cdot \max\{(5ps)^s, (5/64)^s\} + \varepsilon_0 \cdot 2^{(s+1)(3s+\log M)}\right),$$

all "switchings" were successful and we got a circuit of depth-2, size at most $M \cdot 2^s$, bottom fan-in s and a top OR gate, i.e. we got an s-**DNF**.

As for the second item of the theorem, observe that θ is selected by sampling $d-1$ binary strings from $\mathcal{G}$: one string w_1 of length $n \cdot \log_2(64)$ (consisting of n blocks of $\log_2(64)$ bits each) and $(d-2)$ strings, $w_2, \ldots, w_{d-1}$, of length $n \cdot q$ (consisting of n blocks of q bits each). We denote the concatenation of these $d-1$ strings by w. The i-th bit in θ is fixed (i.e. $\theta_i = \square$) iff the i-th block in one of the $d-1$ strings contains a zero. Thus, the event $\theta(i) = \square$ may be expressed as an OR of $6 + (d-2)q$ literals over w. The event that a set of T variables is fixed may be expressed as a **CNF** of size $|T| \leq n$ in the bits of w. By Corollary 55, the distribution of w is $\varepsilon_0 \cdot (d-1)$-pseudorandom for **CNFs** of size at most $M \cdot 2^s \cdot 2^{s(q+1)}$ and in particular to **CNFs** of size at most n. Thus,

$$\begin{aligned}\Pr_{w \text{ pseudo-random}}[T \text{ is fixed under } w] &\leq \Pr_{w \text{ random}}[T \text{ is fixed under } w] + \varepsilon_0 \cdot (d-1) \\ &= (1 - p^{d-2}/64)^{|T|} + \varepsilon_0 \cdot (d-1)\,. \end{aligned}$$

◀

▶ **Theorem 57** ([35, Theorem 12, restated slightly]). *Let C be a size M, depth d circuit, and $\varepsilon > 0$. Then, there exists a pseudorandom generator $\mathcal{G}_1$ of seed length $\tilde{O}(\log^3(M/\varepsilon))$ such that:*

- $|\boldsymbol{Pr}_{\rho\sim\mathcal{G}_1, x\sim U_n}[C_\rho(x) = 1] - \boldsymbol{Pr}_{y\sim U_n}[C(y) = 1]| < \varepsilon$.
- *Let p be the largest power of $1/2$ less than $1/(64\log(8M/\varepsilon))$. Then, each set of variables $T \subseteq [n]$ has probability at most $(1 - p^{d-2}/64)^{|T|} + \varepsilon_0 \cdot (d-1)$ of being unassigned by ρ.*

Proof. We initiate the generator from Thm. 56 based on the generator from Thm. 52. We choose parameters so that the bound we get from Thm. 56 is at most $\varepsilon/2$. Choosing s to be a power of 2 between $\log(8M/\varepsilon)$ to $2\log(8M/\varepsilon)$, $p = 1/64s$ and $\varepsilon_0 = 2^{-9s^2}$ guarantees that

$$M \cdot (2^{2s+1} \cdot \max\{(5ps)^s, (5/64)^s\} + \varepsilon_0 \cdot 2^{(s+1)\cdot(3s+\log M)}) \leq \varepsilon/2\,.$$

The choice also guarantees that $\varepsilon_0 \leq \varepsilon/2$. In order to apply Thm. 56 with these parameters, the generator $\mathcal{G}$ in Thm. 52 should ε_0-fool circuits of size $M' = M \cdot 2^{(q+2)s} = 2^{O(\log(M/\varepsilon)\log\log(M/\varepsilon))}$. Theorem 52 guarantees that seed-length $r = \tilde{O}(\log(M') \cdot \log(M'/\varepsilon_0)) = \tilde{O}(\log^3(M/\varepsilon))$ is enough.

The generator in Thm. 56, $\mathcal{G}_0$, selects a set of coordinates $J = \{i \in [n] : \theta(i) = *\}$. Thm. 56 guarantees that with probability at least $(1 - \varepsilon/2)$ over the choice of J and the restriction of J^c by random bits, C reduces to an s-**DNF** of size at most $M \cdot 2^s$. We then assign values to the variables indexed by J according to De et al. generator, $\mathcal{G}$. Overall, we need seed-length $(d-1) \cdot r + r = dr$, where the first term comes from sampling from $\mathcal{G}_0$ and the second from sampling according to $\mathcal{G}$.

For any fixed choice of θ we have:

$$\Pr_{y\sim U_n}[C(y) = 1] = \Pr_{x\sim U_J, z\sim U_{J^c}}[C(x,z) = 1] = \mathop{\mathbf{E}}_{z}[\Pr_{x}[C(x,z) = 1]]\,.$$

Hence also for $\mathcal{G}_0$ which is a distribution over selections θ the following holds

$$\Pr_{y\sim U_n}[C(y) = 1] = \mathop{\mathbf{E}}_{\theta}\mathop{\mathbf{E}}_{z}[\Pr_{x}[C(x,z) = 1]]$$

For choices (θ, z) such that $C_z(x) := C(x, z)$ is an s-**DNF** of size at most $M \cdot 2^s$, we have $\mathbf{Pr}_{x\sim U_J}[C(x,z)=1] = \mathbf{Pr}_{x\sim\mathcal{G}}[C(x,z)=1] \pm \varepsilon_0$. In the case where $C_z(x)$ is not an s-**DNF** of size $M \cdot 2^s$ we trivially have $\mathbf{Pr}_{x\sim U_J}[C(x,z)=1] = \mathbf{Pr}_{x\sim\mathcal{G}}[C(x,z)=1] \pm 1$. However, this is a rare event that happens with probability at most $\varepsilon/2$. Overall, we have

$$\mathop{\mathbf{Pr}}_{y\sim U_n}[C(y)=1] = \mathop{\mathbf{E}}_{\theta}\mathop{\mathbf{E}}_{z}[\mathop{\mathbf{Pr}}_{x\sim U_J}[C(x,z)=1]] = \mathop{\mathbf{E}}_{\theta}\mathop{\mathbf{E}}_{z}[\mathop{\mathbf{Pr}}_{x\sim\mathcal{G}}[C(x,z)=1]] \pm (\varepsilon_0 + \varepsilon/2)\ .$$

which completes the proof of the first item as $\varepsilon_0 \le \varepsilon/2$.

Note that the generator $\mathcal{G}_1$ selects J and assigns values to the variables in J. It does not assign any of the variables in J^c. In this way, Trevisan and Xue change roles between the fixed and alive parts of the restriction: starting with pseudorandom restriction where J^c is fixed randomly and J is kept alive, they end up with a pseudorandom restriction where J is fixed pseudorandomly and J^c is kept alive.

The second item follows by observing that a set of variables is fixed in Thm. 56 iff it is unassigned here. ◀

▶ **Theorem 58** ([35, Theorem 13, improved]). *For every M, d, n, ε there is a polynomial time computable ε-pseudorandom generator for circuits of size M and depth d on n variables, whose seed length is $\tilde{O}(\log^{d+1}(M/\varepsilon)\cdot\log n)$.*

Proof. If $d \le 2$ we apply Thm. 52. Otherwise, we may assume $d \ge 3$. As in Thm. 57, let p be the largest power of $1/2$ which is smaller than $1/(64\log(8M/\varepsilon))$. Let $p' = p^{d-2}/64$. The theorem follows by applying $R = 3\ln(n)/p'$ independent random restrictions from $\mathcal{G}_1$, each with parameter $\varepsilon/2R$. Let $t = \log(2/\varepsilon)$, and let $T \subseteq [n]$ be a set of size t. The probability T remains totally unfixed after R iterations is at most

$$((1-p')^t + \varepsilon_0\cdot(d-1))^{3\ln(n)/p'}$$

where recall that $\varepsilon_0 = 2^{-\Omega(\log^2(M/\varepsilon))}$. We have $(1-p')^t > 1 - p't \ge 1 - \frac{\log(2/\varepsilon)}{64\log(8/\varepsilon)} > 1/2$, so

$$(1-p')^t + \varepsilon_0\cdot(d-1) \le (1-p')^t\cdot(1+2\varepsilon_0\cdot(d-1))\ ,$$

and we get

$$\begin{aligned}((1-p')^t + \varepsilon_0\cdot(d-1))^{3\ln(n)/p'} &< \left((1-p')^t\cdot(1+2\varepsilon_0\cdot(d-1))\right)^{3\ln(n)/p'} \\ &\le e^{(-p't)\cdot 3\ln(n)/p'}\cdot e^{2\varepsilon_0\cdot(d-1)\cdot 3\ln(n)/p'} \qquad (1+x\le e^x) \\ &= n^{-3t}\cdot e^{o(1)} \ = O(n^{-3t})\ .\end{aligned}$$

As there are only at most n^t sets T of size t, applying union bound, with probability at most $O(n^{-2t}) < \varepsilon/2$ there exists a set of size t which is unassigned. In the complement event, there are less than t variables alive, and we may sample from a t-wise independent distribution to fool the remaining circuit. Overall the seed length is

$$\left(\frac{3\ln(n)}{p'}\cdot\tilde{O}(\log^3(M/\varepsilon)\right) + O\left(\log(1/\varepsilon)\cdot\log n\right) = \tilde{O}(\log^{d+1}(M/\varepsilon)\cdot\log n)\ .$$ ◀

Trading Information Complexity for Error*

Yuval Dagan[1], Yuval Filmus[2], Hamed Hatami[3], and Yaqiao Li[4]

1 **Technion – Israel Institute of Technology, Haifa, Israel**
`yuval.dagan@cs.technion.ac.il`
2 **Technion – Israel Institute of Technology, Haifa, Israel**
`yuvalfi@cs.technion.ac.il`
3 **McGill University, Montreal, QC, Canada**
`hatami@cs.mcgill.ca`
4 **McGill University, Montreal, QC, Canada**
`yaqiao.li@mail.mcgill.ca`

Abstract

We consider the standard two-party communication model. The central problem studied in this article is how much can one save in information complexity by allowing an error of ϵ.

- For arbitrary functions, we obtain lower bounds and upper bounds indicating a gain that is of order $\Omega(h(\epsilon))$ and $O(h(\sqrt{\epsilon}))$. Here h denotes the binary entropy function.
- We analyze the case of the two-bit AND function in detail to show that for this function the gain is $\Theta(h(\epsilon))$. This answers a question of Braverman et al. [4].
- We obtain sharp bounds for the set disjointness function of order n. For the case of the distributional error, we introduce a new protocol that achieves a gain of $\Theta(\sqrt{h(\epsilon)})$ provided that n is sufficiently large. We apply these results to answer another of question of Braverman et al. regarding the randomized communication complexity of the set disjointness function.
- Answering a question of Braverman [3], we apply our analysis of the set disjointness function to establish a gap between the two different notions of the prior-free information cost. In light of [3], this implies that amortized randomized communication complexity is not necessarily equal to the amortized distributional communication complexity with respect to the hardest distribution.

As a consequence, we show that the ϵ-error randomized communication complexity of the set disjointness function of order n is $n[C_{\mathrm{DISJ}} - \Theta(h(\epsilon))] + o(n)$, where $C_{\mathrm{DISJ}} \approx 0.4827$ is the constant found by Braverman et al. [4].

1998 ACM Subject Classification F.1.m [Computation by Abstract Devices] Miscellaneous

Keywords and phrases communication complexity, information complexity

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.16

1 Introduction

Communication complexity studies the amount of communication needed to compute a function whose inputs are spread among several parties. It has many applications to different areas of complexity theory and beyond, mostly as a technical tool used for proving lower bounds. Traditionally, communication complexity has been studied through a combinatorial lens. Recently, a new approach to communication complexity via information theory has arisen, forming the area of *information complexity* [10, 1, 2]. While communication complexity is concerned with minimizing the amount of communication required for two players to

* H. H. was supported by NSERC.

COMPUTATIONAL COMPLEXITY CONFERENCE

evaluate a function, information complexity is concerned with the amount of information that the communicated bits reveal about the players' inputs.

The study of information complexity is motivated by fundamental questions regarding compressing communication [2, 6, 3, 15] that extend the seminal work of Shannon [28] to the setting where interaction is allowed. Moreover, it has important applications to communication complexity, and in particular to the study of the direct-sum problem [1, 10, 17, 8, 7], a problem that has been studied extensively in the past [13, 10, 19, 16, 2, 21, 17, 18, 8, 7]. For example, the only known direct-sum result for general randomized communication complexity is proven via information-theoretic techniques in [2].

One of the most spectacular applications of information complexity, due to Braverman et al. [4], is determining the exact first order communication complexity of set disjointness. Set disjointness is one of the most important functions in communication complexity, and as a result it has been studied extensively in the past four decades (see the surveys [11, 29] and the references therein). In this communication problem, which is denoted by DISJ_n, Alice and Bob each receives a subset of $\{1, \ldots, n\}$ and their goal is to determine whether their sets are disjoint or not. The goal is to determine the asymptotc rate of growth of the *randomized communication complexity* $R_\epsilon(\mathrm{DISJ}_n)$ of set disjointness, defined as the smallest number of bits exchanged by the two players in a protocol which computes the function correctly with probability at least $1 - \epsilon$ on every input. The correct asymptotic $R_\epsilon(\mathrm{DISJ}_n) = \Theta(n)$ was first proved by Kalyanasundaram and Schnitger [20]. Although later Razborov [26] gave a shorter proof, still despite several decades of research in this area, all known proofs for this fact are intricate and sophisticated. It was thus a great breakthrough when Braverman et al. determined the exact constant in the asymptotics of $R_\epsilon(\mathrm{DISJ}_n)$ as $\epsilon \to 0$ by employing several recent results from the area of information complexity. They proved that as the error parameter ϵ tends to 0, the quantity $\lim_{n\to\infty} R_\epsilon(\mathrm{DISJ}_n)/n$ tends to a constant $C_{\mathrm{DISJ}} \approx 0.4827$.

Our major result determines the asymptotic rate of growth of $R_\epsilon(\mathrm{DISJ}_n)$ for *constant* $\epsilon \leq 1/2$:

$$\lim_{n\to\infty} \frac{R_\epsilon(\mathrm{DISJ}_n)}{n} = C_{\mathrm{DISJ}} - \Theta(h(\epsilon)). \tag{1}$$

As in the work of Braverman et al., we obtain our result by analyzing the information complexity of the 2-bit AND function (in which each player gets one bit). Roughly speaking, the *information complexity* $\mathrm{IC}_\mu(f, \epsilon)$ of a function f with respect to a distribution μ on the inputs is the minimal amount of information that the players need to leak in any protocol that computes f correctly with probability at least $1 - \epsilon$ on every input[1]. The asymptotic estimate on $R_\epsilon(\mathrm{DISJ}_n)$ follows by analyzing $\mathrm{IC}^0(\mathrm{AND}, \epsilon) := \min \mathrm{IC}_\mu(\mathrm{AND}, \epsilon)$, where the minimum is taken over all distributions μ such that $\mu(1, 1) = 0$. Specifically, we prove the following bound:

$$\mathrm{IC}^0(\mathrm{AND}, \epsilon) = C_{\mathrm{DISJ}} - \Theta(h(\epsilon)), \tag{2}$$

where the upper bound is attained by a protocol having one-sided error (only allowed to make a mistake on the input $(1, 1)$). The upper bound follows from a black-box modification

[1] There are two different ways to measure information leakage. The usual notion, internal information complexity, measures how much each player learns about the other player's input. External information complexity, studied in this paper only in passing, measures how much an external observer learns about the players' input.

of the optimal protocol for AND found by Braverman et al. The lower bound is significantly harder, requiring several novel ideas which could have wider applicability. We sketch these ideas later on in the introduction.

It is natural to ask whether a bound of the form (2) holds for arbitrary functions f. Braverman et al. [4] considered this question in the context of distributional information complexity[2]. The *distributional information complexity* $\mathrm{IC}_\mu(f,\mu,\epsilon)$ of a function f with respect to a distribution μ on the inputs is the minimal amount of information that the players need to leak in any protocol that computes f correctly with probability at least $1-\epsilon$ *when the inputs are drawn according to* μ. They showed that $\mathrm{IC}_\mu(f,\mu,\epsilon) \geq \mathrm{IC}_\mu(f,\mu,0) - O(h(\epsilon^{1/8}))$ (here and below, the hidden constant depends on f and μ). We significantly improve this lower bound, and obtain the first non-trivial upper and lower bounds for general functions:

$$\mathrm{IC}_\mu(f,\mu,0) - O(h(\sqrt{\epsilon})) \leq \mathrm{IC}_\mu(f,\mu,\epsilon) \leq \mathrm{IC}_\mu(f,\mu,0) - \Omega(h(\epsilon)),$$
$$\mathrm{IC}_\mu(f,0) - O(h(\sqrt{\epsilon})) \leq \mathrm{IC}_\mu(f,\epsilon) \leq \mathrm{IC}_\mu(f,0) - \Omega(h(\epsilon)).$$

Our results hold in both the non-distributional and distributional settings, as well as in the prior-free settings explained below. The upper bounds use the same black-box technique used to prove the upper bound in (2). The lower bounds use *protocol completion*, a novel technique which also figures in the proof of the lower bound in (2).

In classical communication complexity, the distributional setting arises from an application of Yao's minimax principle: $R_\epsilon(f)$ is the maximum over μ of the communication complexity of deterministic protocols which compute f correctly with probability at least $1-\epsilon$ when the inputs are drawn according to μ. This connection suggests searching for an analog of $R_\epsilon(f)$ in the setting of information complexity. Braverman [3] defined two such notions of *prior-free information complexity*: $\mathrm{IC}(f,\epsilon) = \max_\mu \mathrm{IC}_\mu(f,\epsilon)$, and $\mathrm{IC}^D(f,\epsilon) = \max_\mu \mathrm{IC}_\mu(f,\mu,\epsilon)$. Using the minimax theorem, he showed that the two notions coincide when $\epsilon = 0$. He conjectured that the two notions coincide for all ϵ, but he could only prove the following bound, for $0 < \alpha < 1$:

$$\mathrm{IC}^D(f,\epsilon) \leq \mathrm{IC}(f,\epsilon) \leq \frac{\mathrm{IC}^D(f,\alpha\epsilon)}{1-\alpha}.$$

We separate the two notions of prior-free information complexity, thus showing that this tradeoff is essentially optimal for set disjointness:

$$\frac{\mathrm{IC}^D(\mathrm{DISJ}_n,\epsilon)}{n} \lesssim C_{\mathrm{DISJ}} - \Theta(\sqrt{h(\epsilon)}),$$
$$\frac{\mathrm{IC}(\mathrm{DISJ}_n,\epsilon)}{n} \geq C_{\mathrm{DISJ}} - \Theta(h(\epsilon)),$$

where $\lesssim$ hides a $o_n(1)$ term. The upper bound on $\mathrm{IC}^D(\mathrm{DISJ}_n,\epsilon)$ follows from a novel protocol for set disjointness which is asymptotically optimal in the distributional prior-free setting, while the lower bound on $\mathrm{IC}(\mathrm{DISJ}_n)$ follows from the proof of (1).

Since information complexity is amortized communication complexity, we can also state our separation in terms of communication complexity. Let $R^m_\epsilon(f^m)$ denote the randomized communication complexity of computing m copies of f with an error of at most ϵ on each of the m inputs. Similarly, let $D^{\mu,m}_\epsilon(f^m)$ denote the corresponding distributional notion, where

[2] Information complexity and distributional information complexity are often confused in the literature. One reason might be that they are the same in the zero-error prior-free setting, as shown by Braverman [3] and explained further below.

the error is measured when the inputs are drawn according to μ. Braverman [3] showed that $\mathrm{IC}(f,\epsilon) = \lim_{m\to\infty} R^m_\epsilon(f^m)/m$ and $\mathrm{IC}^D(f,\epsilon) = \lim_{m\to\infty} \max_\mu D^{\mu,m}_\epsilon(f^m)/m$, and so our separation of $\mathrm{IC}(\mathrm{DISJ}_n,\epsilon)$ and $\mathrm{IC}^D(\mathrm{DISJ}_n,\epsilon)$ also separates $\max_\mu D^{\mu,m}_\epsilon(\mathrm{DISJ}^m_n)$ and $R^m_\epsilon(\mathrm{DISJ}^m_n)$.

Finally, given a function f we characterize all measures μ such that $\mathrm{IC}_\mu(f,0) = 0$. We also prove a few results about external information complexity IC^{ext} (which we do not define here). Given a function f we characterize all measures μ such that $\mathrm{IC}^{ext}_\mu(f,0) = 0$. We also show that the upper bound $\mathrm{IC}_\mu(f,\epsilon) \le \mathrm{IC}_\mu(f,0) - \Omega(h(\epsilon))$ fails for external information complexity: $\mathrm{IC}^{ext}_\mu(\mathrm{XOR},\epsilon) \ge \mathrm{IC}^{ext}_\mu(\mathrm{XOR},0) - 3\epsilon$, where the distribution μ is given by $\mu(0,0) = \mu(1,1) = 1/2$.

1.1 Techniques

1.1.1 Stability for the buzzer protocol

At the heart of the lower bound $\mathrm{IC}^0(\mathrm{AND},\epsilon) \ge C_{\mathrm{DISJ}} - O(h(\epsilon))$ lies a stability result for almost-optimal protocols for AND.

Braverman et al. [4] gave an optimal protocol for the AND function, which they call the *buzzer protocol*. They also showed that this protocol is essentially the *unique* optimal protocol for the AND function. We prove a stability version of this result: *any ϵ-error protocol for AND whose information cost is close to that of the buzzer protocol must be similar to the buzzer protocol*.

There are many possible notions of similarity, and ours (for reasons that will become clear below) focuses on the *leaf distribution* of the protocol, which is the distribution of the terminal point of the protocol. Our stability result roughly states that any ϵ-error protocol for AND whose information cost is close to that of the buzzer protocol must have a leaf distribution which is similar to the leaf distribution of the buzzer protocol.

We prove our stability result by strengthening the technique of *local concavity constraints* introduced by Braverman et al. On the way, we also simplify the arguments of Braverman et al. by replacing the discrete second derivatives used by Braverman et al. with their continuous counterparts.

1.1.2 The buzzer protocol as a random walk

One of our main insights is an alternative description of the buzzer protocol as a random walk.

As part of their analysis of the AND function, Braverman et al. introduced a new perspective on communication protocols, viewing a communication protocol as a random walk on the space of distributions. Given an initial distribution over the inputs, they associate with each node in the protocol tree the a posteriori distribution of the inputs, which is the distribution of the inputs given that the protocol arrives at the node. Instead of walking down the protocol tree, we can think of the protocol as a random walk on these a posteriori input distributions.

Braverman et al. describe the buzzer protocol as a continuous time protocol which ends abruptly when one of the players buzzes. We give an alternative description of the buzzer protocol, as a random walk on the space of distributions. Consider the case in which the input distribution μ is a product distribution given by $\Pr[X = 1] = p$ and $\Pr[Y = 1] = q$, where X, Y are the input bits of Alice and Bob, respectively; we denote this distribution succinctly by (p,q). The buzzer protocol is the limit $\epsilon \to 0$ of a random walk which starts at (p,q), and at each step moves either vertically or horizontally depending on the current

distribution (a, b): if $a \geq b$ it moves to $(a, b + \epsilon)$ or to $(a, b - \epsilon)$, with probability $1/2$ each, and if $a < b$ it moves to $(a + \epsilon, b)$ or to $(a - \epsilon, b)$, with probability $1/2$ each. In both cases we clip the protocol to $[0, 1]^2$. The random walk terminates when $a = 0$ or $b = 0$, in which case it outputs 0, and when $a = b = 1$, in which case it outputs 1.

Our description of the buzzer protocol has two main advantages over the original one. First, the a posteriori distribution varies continuously in our protocol. In contrast, in the original description the a posteriori distribution "collapses" when one of the players presses the buzzer. Second, our protocol is the same for all distributions, whereas the original buzzer protocol has an additional symmetrization step to handle asymmetric initial distributions. Both of these properties simplify our analysis.

1.1.3 Product parametrization

Our most important technical innovation is a way of analyzing non-product distributions as if they were product distributions. Since product distributions are often much easier to analyze, we believe this idea could have many further applications, which we hope to explore in future work.

So far we have described the buzzer protocol as a random walk only when the initial distribution is a product distribution. In that case, the random walk is supported on the manifold of product distributions. More generally, for any initial distribution μ, all reachable a posteriori distributions can be obtained from μ by scaling the rows and columns. Therefore the manifold of distributions reachable from μ, which we call the *μ-manifold*, can be parametrized by product distributions. This key idea allows us to treat any initial distribution μ as if it were a product distribution, as we now explain in detail.

The information cost of a protocol equals the difference between the amount of information not known to the players before it begins, and the expected information not known after it ends. The information cost can easily be calculated given the second term, which is known as the *concealed information*. The concealed information can be viewed as the expected reward (corresponding to unrevealed information) obtained at the leaves of the protocol. Finding a protocol that minimizes the information cost is thus equivalent to finding a random walk that maximizes the expected reward.

Using the product parametrization, we can convert a random walk on the μ-manifold to a random walk on the manifold of product distributions. The concealed information is replaced by the *scaled concealed information*, which also equals some expected reward over the leaves of the protocol. The concealed information, hence the information cost, can easily be extracted from this parameter. This allows us to analyze protocols on general input distributions as if the input distribution were a product distribution, the only difference being the scaling of concealed information at the leaves.

While we only use this technique for analyzing the AND function, it applies to general functions on general input domains. We believe that this technique has wide applicability in the area of information complexity, since product distributions are often easier to analyze than general distributions.

1.1.4 Protocol completion

We prove the lower bounds on $\mathrm{IC}_\mu(f, \epsilon)$ and on $\mathrm{IC}^0(\mathrm{AND}, \epsilon)$ using the technique of *protocol completion*. Given an ϵ-error protocol for f, we complete it to a zero-error protocol for f in a natural way: when the protocol terminates at a posterior distribution ν (which is the distribution of the inputs given the transcript of the protocol and the initial distribution μ),

we run a zero-error protocol for f which is information-efficient for the distribution ν. Using the buzzer protocol, we give a protocol for f whose information cost is $O(h(\sqrt{\alpha}))$, where $1-\alpha$ is the probability of the most probable output given ν. Since $\mathbb{E}[\alpha] = \epsilon$, this shows that we can complete the given ϵ-error protocol to a zero-error protocol for f at a cost of $O(h(\sqrt{\epsilon}))$ in the information cost, implying the bound $\mathrm{IC}_\mu(f,\epsilon) + O(h(\sqrt{\epsilon})) \geq \mathrm{IC}_\mu(f,0)$.

For the case $f = \mathrm{AND}$, we are able to improve on this result, tightening the gap from $O(h(\sqrt{\epsilon}))$ to $O(h(\epsilon))$, using the stability result for the buzzer protocol. The product parametrization allows us to consider the posterior distribution ν as a product distribution (a,b). If $\max(a,b) = \Omega(1)$ then the buzzer protocol has information cost $O(h(\alpha))$ rather than just $O(h(\sqrt{\alpha}))$ (recall that $1-\alpha$ is the probability of the most probable output given ν). Suppose now that we are given an ϵ-error protocol π for AND. Our goal is to prove that $\mathrm{IC}_\mu(\pi) \geq \mathrm{IC}_\mu(\mathrm{AND},0) - C_\mu h(\epsilon)$ for some $C_\mu > 0$ (here $\mathrm{IC}_\mu(\pi)$ is the information cost of π). We can complete π to a zero-error protocol π_0 at a cost of $O(h(\sqrt{\epsilon}))$. We can assume that $\mathrm{IC}_\mu(\pi_0) \leq \mathrm{IC}_\mu(\mathrm{AND},0) - C_\mu h(\epsilon) + O(h(\sqrt{\epsilon}))$, and so π_0 is an almost-optimal protocol for AND. Our stability result shows that a random leaf (a,b) of π_0 satisfies $\max(a,b) \geq c_\mu$ with high probability, for some $c_\mu > 0$. It follows that the same holds for π, and so the cost of completion is only $O(h(\epsilon))$.

1.1.5 Black-box modification

We prove the upper bounds on $\mathrm{IC}_\mu(f,\epsilon)$ and (as a special case) on $\mathrm{IC}^0(\mathrm{AND},\epsilon)$ using a simple black-box argument, which modifies an optimal zero-error protocol to a slightly more information-efficient ϵ-error protocol. Given a zero-error protocol π for f, one way to create an ϵ-error protocol for f is to run π with probability $1-\epsilon$, and output some constant value with probability ϵ. However, this only saves $O(\epsilon)$ bits of information. Our modification is different: we identify a player P and two inputs z_0, z_1, and run the following protocol π':

- With probability ϵ (sampled privately by P), if the input of P is z_1 then P changes its input to z_0.
- The players run π on their possibly modified inputs.

This is also an ϵ-error protocol, and for a suitable choice of the parameters, it turns out that it saves $\Omega(h(\epsilon))$ bits of information compared to π.

When the input distribution μ has full support, it is easy to choose the parameters, by finding two inputs $(x_0,y_0),(x_1,y_1)$ which differ on a single coordinate such that $f(x_0,y_0) \neq f(x_1,y_1)$. Such a choice might not exist when μ doesn't have full support, and instead we rely on a rather delicate binary search argument on the set of transcripts.

We can apply this argument to the AND function, showing that $\mathrm{IC}_\mu(\mathrm{AND},\epsilon) \leq C_{\mathrm{DISJ}} - \Omega(h(\epsilon))$. However, when using this result to obtain a protocol for set disjointness, we encounter a difficulty: in order to obtain an ϵ-error protocol for DISJ_n, it seems at first that we need a protocol for AND having error ϵ/n. This would result in a saving of $O(h(\epsilon/n))$ rather than $O(h(\epsilon))$ per coordinate. A similar difficulty was encountered by Molinaro et al. [25] in a similar context, and they overcame it using protocols that abort. In our case there is a simpler solution: we consider ϵ-error protocols for AND which only make one-sided error, outputing 0 when the correct answer is 1 (the black-box argument can be modified to produce such protocols). If we apply such a protocol coordinatewise to compute the intersection of X,Y, then we always compute the intersection correctly when X,Y are disjoint, and we mistakenly compute the intersection to be empty when X,Y are not disjoint with probability at most $\epsilon^{|X\cap Y|} \leq \epsilon$. The resulting protocol thus computes set disjointness correctly with probability at least $1-\epsilon$ on every input.

1.1.6 Computing set disjointness with error

The lower bound $\mathrm{IC}^0(\mathrm{AND}, \epsilon) \geq C_{\mathrm{DISJ}} - O(h(\epsilon))$ implies a similar lower bound on the information complexity of set disjointness: $\mathrm{IC}(\mathrm{DISJ}_n, \epsilon)/n \geq C_{\mathrm{DISJ}} - O(h(\epsilon))$. In contrast, we can save more than $h(\epsilon)$ in the distributional prior-free setting: $\mathrm{IC}^D(\mathrm{DISJ}_n, \epsilon)/n \leq C_{\mathrm{DISJ}} - \Theta(\sqrt{h(\epsilon)}) + o(1)$. A minimax argument of Braverman [3] shows that this bound is tight. We prove this upper bound using a novel protocol for set disjointness. Given a distribution μ, we describe a protocol π which has error ϵ with respect to μ, whose information cost satisfies

$$\mathrm{IC}_\mu(\pi) \leq n[C_{\mathrm{DISJ}} - \Omega(\sqrt{h(\epsilon)})] + O(\log n).$$

Let p be the probability the input sets X, Y are not disjoint, when $(X, Y) \sim \mu$. The protocol proceeds as follows:

- Using public randomness, Alice and Bob sample a permutation σ on $1, \ldots, n$.
- For $i = 1, \ldots, n$, Alice and Bob run a protocol for AND on $X_{\sigma(i)}, Y_{\sigma(i)}$ which has one-sided error $\epsilon/2p$ with respect to the conditional distribution of $X_{\sigma(i)}, Y_{\sigma(i)}$, declaring X, Y to be not disjoint (and halting the protocol) if the AND protocol answers $X_{\sigma(i)} = Y_{\sigma(i)} = 1$.
- Declare X, Y to be disjoint.

The protocol only makes an error when the inputs are not disjoint, and in that case it makes an error with probability $(\epsilon/2p)^{|X \cap Y|} \leq \epsilon/2p$. Since the inputs are non-disjoint with probability p, the overall error probability is $\epsilon/2 < \epsilon$. A tricky but standard argument shows that this protocol saves roughly $\Omega(n\sqrt{h(\epsilon)})$ bits of information.

2 Preliminaries

In this section we introduce some basic notation and facts, and review the necessary background for the paper.

2.1 Notation and basic estimates

We typically denote random variables by capital letters (e.g A, B, C, Π). For the sake of brevity, we shall write $A_1 \ldots A_n$ to denote the random variable $(A_1, \ldots, A_n)$ and *not* the product of the A_i's. We use $[n]$ to denote the set $\{1, \ldots, n\}$, and $\operatorname{supp} \mu$ to denote the support of a measure μ.

For a finite set Ω, we denote by $\Delta(\Omega)$, the set of all discrete probability distributions on Ω. For $\mu, \nu \in \Delta(\Omega)$, we denote their *total variation distance* with

$$|\mu - \nu| := \frac{1}{2} \sum_{x \in \Omega} |\mu(x) - \nu(x)|.$$

For every $\epsilon \in [0, 1]$, $h(\epsilon) = -\epsilon \log \epsilon - (1 - \epsilon) \log(1 - \epsilon)$ denotes the *binary entropy*, where here and throughout the paper $\log(\cdot)$ is in base 2, and $0 \log 0 = 0$.

2.2 Communication complexity

The notion of two-party communication complexity was introduced by Yao [30] in 1979. In this model there are two players (with unlimited computational power), often called Alice and Bob, who wish to collaboratively perform a task such as computing a given function $f \colon \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$. Alice receives an input $x \in \mathcal{X}$ and Bob receives $y \in \mathcal{Y}$. Neither of them knows the other player's input, and they wish to communicate in accordance with an agreed-upon

protocol π to compute $f(x, y)$. The protocol π specifies as a function of (only) the transmitted bits whether the communication is over, and if not, who sends the next bit. Furthermore π specifies what the next bit must be as a function of the transmitted bits, and the input of the player who sends the bit. We will assume that when the protocol terminates Alice and Bob agree on a value as the output of the protocol. We denote this value by $\pi(x, y)$. The *communication cost* of π is the total number of bits transmitted on the worst case input. The *transcript* of an execution of π is a string Π consisting of a list of all the transmitted bits during the execution of the protocol. As protocols are defined using protocol trees, transcripts are in one-to-one correspondence with the leaves of this tree.

In the randomized communication model, the players might have access to a shared random string (*public randomness*), and their own private random strings (*private randomness*). These random strings are independent, but they can have any desired distributions individually. In the randomized model the *transcript* also includes the public random string in addition to the transmitted bits. Similar to the case of deterministic protocols, the *communication cost* is the total number of bits transmitted on the worst case input and random strings. The *average communication cost* of the protocol is the expected number of bits transmitted on the worst case input.

For a function $f\colon \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$ and a parameter $\epsilon > 0$, we denote by $R_\epsilon(f)$ the communication cost of the best randomized protocol that computes the value of $f(x, y)$ correctly with probability at least $1 - \epsilon$ for *every* (x, y).

2.3 Information complexity

The setting is the same as in communication complexity, where Alice and Bob (having infinite computational power) wish to mutually compute a function $f\colon \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$. To be able to measure information, we also need to assume that there is a prior distribution μ on $\mathcal{X} \times \mathcal{Y}$.

For the purpose of communication complexity, once we allow public randomness, it makes no difference whether we permit the players to have private random strings or not. This is because the private random strings can be simulated by parts of the public random string. On the other hand, for information complexity, it is crucial to permit private randomness, and once we allow private randomness, public randomness becomes inessential. Indeed, one of the players can use her private randomness to generate the public random string, and then transmit it to the other player. Although this might have very large communication cost, it has no information cost, as it does not reveal any information about the players' inputs.

Probably the most natural way to define the information cost of a protocol is to consider the amount of information that is revealed about the inputs X and Y to an external observer who sees the transmitted bits and the public randomness. This is called the *external information cost* and is formally defined as the mutual information between XY and the transcript of the protocol (recall that the transcript also contains the public random string). While this notion is interesting and useful, it turns out there is a different way of defining the information cost that enjoys certain desirable properties that the external information cost lack. This is called the *internal information cost* or just the *information cost* for short, and is equal to the amount of information that Alice and Bob learn about each other's inputs from the communication. Note that Bob knows Y, the public randomness R, and his own private randomness R_B, and thus what he learns about X *from the communication* can be measured by the conditional mutual information $I(X; \Pi | YRR_B)$. Similarly, what Alice learns about Y from the communication can be measured by $I(Y; \Pi | XRR_A)$ where R_A is Alice's private random string. It is not difficult to see [2] that conditioning on the public and private randomness does not affect these quantities. In other words $I(X; \Pi | YRR_B) = I(X; \Pi | Y)$ and $I(Y; \Pi | XRR_A) = I(Y; \Pi | X)$. We summarize these in the following definition.

▶ **Definition 1.** The *internal information cost* and the *external information cost* of a protocol π with respect to a distribution μ on inputs from $\mathcal{X} \times \mathcal{Y}$ are defined as

$$\mathrm{IC}_\mu(\pi) = I(\Pi; X|Y) + I(\Pi; Y|X),$$

and

$$\mathrm{IC}^{ext}_\mu(\pi) = I(\Pi; XY),$$

respectively, where $\Pi = \Pi_{XY}$ is the transcript of the protocol when it is executed on $XY \sim \mu$.

We will be interested in certain *communication tasks*. Let $[f, \epsilon]$ denote the task of computing the value of $f(x, y)$ correctly with probability at least $1 - \epsilon$ for *every* (x, y). Thus a protocol π performs this task if

$$\Pr[\pi(x, y) \neq f(x, y)] \leq \epsilon, \quad \forall\ (x, y) \in \mathcal{X} \times \mathcal{Y}.$$

Given another distribution ν on $\mathcal{X} \times \mathcal{Y}$, let $[f, \nu, \epsilon]$ denote the task of computing the value of $f(x, y)$ correctly with probability at least $1 - \epsilon$ if the input (x, y) is sampled from the distribution ν. A protocol π performs this task if

$$\Pr_{(x,y)\sim\nu}[\pi(x, y) \neq f(x, y)] \leq \epsilon.$$

Note that a protocol π performs $[f, 0]$ if it computes f correctly on *every* input while performing $[f, \nu, 0]$ means computing f correctly on the inputs that belong to the support of ν.

We will also need a one-sided version of the task $[f, \epsilon]$. Let $[f, \epsilon, z_1 \to z_0]$ denote the task of computing the value of $f(x, y)$ correctly with probability at least $1 - \epsilon$ for *every* (x, y), allowing the protocol to err only if it outputs z_0 instead of z_1. Thus a protocol π performs this task if it performs the task $[f, \epsilon]$, and additionally

$$\pi(x, y) \neq f(x, y) \implies f(x, y) = z_1 \text{ and } \pi(x, y) = z_0.$$

The *information complexity* of a communication task T with respect to a measure μ is defined as

$$\mathrm{IC}_\mu(T) = \inf_{\pi:\ \pi \text{ performs } T} \mathrm{IC}_\mu(\pi).$$

It is essential here that we use infimum rather than minimum as there are tasks for which there is no protocol that achieves $\mathrm{IC}_\mu(T)$ while there is a sequence of protocols whose information cost converges to $\mathrm{IC}_\mu(T)$. The *external information complexity* of a communication task T is defined similarly. We will abbreviate $\mathrm{IC}_\mu(f, \epsilon) = \mathrm{IC}_\mu([f, \epsilon])$, $\mathrm{IC}_\mu(f, \nu, \epsilon) = \mathrm{IC}_\mu([f, \nu, \epsilon])$, etc. It is important to note that when μ does not have full support, $\mathrm{IC}_\mu(f, \mu, 0)$ can be strictly smaller than $\mathrm{IC}_\mu(f, 0)$.

▶ Remark (A warning regarding notation). In the literature of information complexity it is common to use "$\mathrm{IC}_\mu(f, \epsilon)$" to denote the distributional error case, i.e. what we denote by $\mathrm{IC}_\mu(f, \mu, \epsilon)$. Unfortunately this has become the source of some confusions in the past, as sometimes "$\mathrm{IC}_\mu(f, \epsilon)$" is used to denote both of the distributional error and the point-wise error cases. To avoid ambiguity we distinguish the two cases by using the different notations $\mathrm{IC}_\mu(f, \mu, \epsilon)$ and $\mathrm{IC}_\mu(f, \epsilon)$.

Similar to the fact that the maximal distributional communication complexity over all measures equals the public coin randomized communication complexity (see e.g., [22, Section 3.4]), below we prove a lemma that establishes a similar relation between $\mathrm{IC}_\mu(f,\nu,\epsilon)$ and $\mathrm{IC}_\mu(f,\epsilon)$.

▶ **Lemma 2.** $\mathrm{IC}_\mu(f,\epsilon) = \max_\nu \mathrm{IC}_\mu(f,\nu,\epsilon)$ *holds for all* $\epsilon \geq 0$.

Note that the maximum exists due to continuity of $\mathrm{IC}_\mu(f,\nu,\epsilon)$ with respect to ν, a fact that is discussed later in Section 2.4 (For $\epsilon = 0$ one can take any full-support ν).

Proof. We only need to show $\mathrm{IC}_\mu(f,\epsilon) \leq \max_\nu \mathrm{IC}_\mu(f,\nu,\epsilon)$ as the other direction is obvious. The proof is an application of von Neumann's minimax theorem.

Pick a small $\delta > 0$, let $C_\delta = \{\pi : \mathrm{IC}_\mu(\pi) \leq \mathrm{IC}_\mu(f,\epsilon) - \delta\}$. Although C_δ is an infinite set, we can approximate it by a finite set by considering only the protocols with bounded communication cost that use only a bounded number of unbiased random bits. This process does not affect the validity of the proof, and hence the minimax theorem is still applicable.

Consider a two-player zero-sum game in which Alice chooses a protocol $\pi \in C_\delta$ and Bob chooses an input $(x,y) \in \mathcal{X} \times \mathcal{Y}$, and define the utility for Alice to be $\Pr[\pi(x,y) = f(x,y)]$. Note that a mixed strategy for Alice is still just a protocol, and a mixed strategy for Bob corresponds to a probability measure on $\mathcal{X} \times \mathcal{Y}$. By our definition of C_δ and the minimax theorem, we have

$$\min_\nu \max_\pi \mathop{\mathbb{E}}_{(x,y)\sim\nu} \Pr[\pi(x,y) = f(x,y)] = \max_\pi \min_\nu \mathop{\mathbb{E}}_{(x,y)\sim\nu} \Pr[\pi(x,y) = f(x,y)] = 1-\epsilon-t(\delta) < 1-\epsilon,$$

where $t(\delta) > 0$ is a positive quantity. This means that there exists a measure ν^*_δ such that for all $\pi \in C_\delta$, $\mathbb{E}_{(x,y)\sim\nu^*_\delta} \Pr[\pi(x,y) \neq f(x,y)] > \epsilon$. Letting $\delta \to 0$ gives $\max_\nu \mathrm{IC}_\mu(f,\nu,\epsilon) \geq \mathrm{IC}_\mu(f,\epsilon)$ as desired. ◀

Finally let us recall the two definitions of the prior-free notions of information complexity introduced in [3]. The *max-distributional information complexity* of a function $f\colon \mathcal{X}\times\mathcal{Y} \to \mathcal{Z}$ is defined as

$$\mathrm{IC}^D(f,\epsilon) = \max_\mu \mathrm{IC}_\mu(f,\mu,\epsilon).$$

The information complexity of f with error ϵ is defined as

$$\mathrm{IC}(f,\epsilon) = \inf_\pi \max_\mu \mathrm{IC}_\mu(\pi),$$

where the infimum is over all protocols π that perform the task $[f,\epsilon]$. It is possible [3] to use a minimax argument and the concavity of $\mathrm{IC}_\mu(\pi)$ with respect to μ to show that

$$\mathrm{IC}(f,\epsilon) = \inf_\pi \max_\mu \mathrm{IC}_\mu(\pi) = \max_\mu \inf_\pi \mathrm{IC}_\mu(\pi) = \max_\mu \mathrm{IC}_\mu(f,\epsilon) = \max_{\mu,\nu} \mathrm{IC}_\mu(f,\nu,\epsilon),$$

where the last equality follows from Lemma 2.

2.4 The continuity of information complexity

It is shown in [5, Lemma 4.4] that for every communication task T, $\mathrm{IC}_\mu(T)$ is uniformly continuous with respect to μ. More precisely, for every two measures μ_1 and μ_2 with $|\mu_1 - \mu_2| \leq \delta$ (the distance is in total variation distance), we have

$$|\mathrm{IC}_{\mu_1}(T) - \mathrm{IC}_{\mu_2}(T)| \leq 2\log(|\mathcal{X}\times\mathcal{Y}|)\delta + 2h(2\delta). \tag{3}$$

The information complexity functions $\mathrm{IC}_\mu(f,\epsilon)$ and $\mathrm{IC}_\mu(f,\nu,\epsilon)$ are both continuous with respect to ϵ. The following simple lemma from [3] proves continuity for $\epsilon \in (0,1]$. The continuity at 0 is more complicated and is proven in [4] (See also Theorem 7 and Theorem 8 below).

▶ **Lemma 3.** *[3] For every $f\colon \mathcal{X}\times\mathcal{Y}\to\mathcal{Z}$, $\epsilon_2>\epsilon_1>0$ and measures μ,ν on $\mathcal{X}\times\mathcal{Y}$, we have*

$$\mathrm{IC}_\mu(f,\nu,\epsilon_1)-\mathrm{IC}_\mu(f,\nu,\epsilon_2)\le(1-\epsilon_1/\epsilon_2)\log|\mathcal{X}\times\mathcal{Y}|, \tag{4}$$

and

$$\mathrm{IC}_\mu(f,\epsilon_1)-\mathrm{IC}_\mu(f,\epsilon_2)\le(1-\epsilon_1/\epsilon_2)\log|\mathcal{X}\times\mathcal{Y}|. \tag{5}$$

Proof. Consider a protocol π with information cost I, and error $\epsilon_2>0$. Here we can consider the distributional error as in (4) or the point-wise error as in (5). Set $\delta=1-\epsilon_1/\epsilon_2$, and let τ be the protocol that with probability $1-\delta$ runs π, and with probability δ Alice and Bob exchange their inputs and compute $f(x,y)$ correctly. The theorem follows as the new protocol has error at most $(1-\delta)\epsilon_2=\epsilon_1$, and information cost at most $I+\delta\log|\mathcal{X}\times\mathcal{Y}|$. ◀

Note that $\mathrm{IC}_\mu(f,\mu,0)$ is not always continuous with respect to μ. For example, let the matrices

$$\mu_\epsilon=\begin{pmatrix}\frac{1-\epsilon}{3} & \frac{1-\epsilon}{3}\\ \frac{1-\epsilon}{3} & \epsilon\end{pmatrix},\qquad \mu=\lim_{\epsilon\to0}\mu_\epsilon=\begin{pmatrix}\frac13 & \frac13\\ \frac13 & 0\end{pmatrix}. \tag{6}$$

represent distributions on $\{0,1\}^2$. Here the entry at the i-th row and j-th column corresponds to the measure of the point $(i-1,j-1)\in\{0,1\}^2$. Now for the 2-bit AND function, we have $\mathrm{IC}_\mu(\mathrm{AND},\mu,0)=0$, while $\mathrm{IC}_{\mu_\epsilon}(\mathrm{AND},\mu_\epsilon,0)=\mathrm{IC}_{\mu_\epsilon}(\mathrm{AND},0)$ as μ_ϵ has full support. Thus

$$\lim_{\epsilon\to0}\mathrm{IC}_{\mu_\epsilon}(\mathrm{AND},\mu_\epsilon,0)=\lim_{\epsilon\to0}\mathrm{IC}_{\mu_\epsilon}(\mathrm{AND},0)=\mathrm{IC}_\mu(\mathrm{AND},0),$$

which is known to be bounded away from 0.

Finally, note that Lemma 3 also implies the continuity of $\mathrm{IC}_\mu(f,\nu,\epsilon)$ with respect to ν when $\epsilon>0$. Indeed if $|\nu_1-\nu_2|\le\delta\le\epsilon$, then a protocol that has distributional error ϵ with respect to ν_2, will have error at most $\epsilon+\delta$ and at least $\epsilon-\delta$ with respect to ν_1. Thus

$$\mathrm{IC}_\mu(f,\nu_1,\epsilon+\delta)\le\mathrm{IC}_\mu(f,\nu_2,\epsilon)\le\mathrm{IC}_\mu(f,\nu_1,\epsilon-\delta). \tag{7}$$

which establishes the desired continuity. A similar example to (6) shows that $\mathrm{IC}_\mu(f,\nu,0)$ is not necessarily continuous with respect to ν.

2.5 Communication protocols as random walks on $\Delta(\mathcal{X}\times\mathcal{Y})$

Recall that $\Delta(\mathcal{X}\times\mathcal{Y})$ denotes the set of probability distributions on $\mathcal{X}\times\mathcal{Y}$. Consider a protocol π and a prior distribution μ on the set of inputs $\mathcal{X}\times\mathcal{Y}$. Suppose that in the first round Alice sends a random signal B to Bob. We can interpret this as a random update of the prior distribution μ to a new distribution $\mu_0=\mu|_{B=0}$ or $\mu_1=\mu|_{B=1}$ depending on the value of B. It is not difficult to see that $\mu_b(x,y)=p_b(x)\mu(x,y)$ for $b=0,1$, where $p_b(x)=\frac{\Pr[B=b|x]}{\Pr[B=b]}$. In other words, μ_b is obtained by multiplying the rows of μ by non-negative numbers. From the law of total expectation,

$$\mu=\mathop{\mathbb{E}}_B[\mu|B]=\Pr[B=0]\mu_0+\Pr[B=1]\mu_1. \tag{8}$$

Similarly if Bob is sending a message, then μ_b is obtained by multiplying the columns of μ by the numbers $p_b(y) = \frac{\Pr[B=b|y]}{\Pr[B=b]}$. That is $\mu_b(x,y) = \mu(x,y)p_b(y)$.

The opposite direction is also true: given a distribution μ, distributions μ_0, μ_1, and $0 \leq p_0, p_1 \leq 1$ such that

- $p_0 + p_1 = 1$,
- μ_0 and μ_1 are obtained from μ by scaling its rows,
- $\mu = p_0\mu_0 + p_1\mu_1$,

one can define a random bit B that can be sent by Alice such that μ_b is μ conditioned on $B = b$ for $b \in \{0,1\}$, and $p_b = \Pr[B = b]$. A similar statement holds for the case where μ_0 and μ_1 are obtained from μ by scaling its columns and B is a signal that will be sent by Bob.

Therefore, we can think of a protocol as a random walk on $\Delta(\mathcal{X} \times \mathcal{Y})$ that starts at μ, and every time that a player sends a message, it moves to a new distribution. Equation (8) implies that this random walk is without drift.

Let Π denote the transcript of the protocol. Note that when the protocol terminates, the random walk stops at $\mu_\Pi := \mu|_\Pi$. Since Π itself is a random variable, μ_Π is a random variable that takes values in $\Delta(\mathcal{X} \times \mathcal{Y})$. Interestingly, both the internal and external information costs of the protocol depend only on the distribution of μ_Π (this is a distribution on the set $\Delta(\mathcal{X} \times \mathcal{Y})$, which itself is a set of distributions) [9]. It does not matter how different the steps of two protocols are, and as long as they both yield the same distribution on $\Delta(\mathcal{X} \times \mathcal{Y})$, they have the same internal and external information cost. Consequently, one can directly work with this random walk, instead of working with the actual protocols.

In order to study the relation between the information complexity and the distribution of μ_Π, define the *concealed information* and *external concealed information* of a protocol π with respect to μ, respectively, as

$$\mathrm{CI}_\mu(\pi) = H(X|\Pi Y) + H(Y|\Pi X) = H(X|Y) + H(Y|X) - \mathrm{IC}_\mu(\pi), \tag{9}$$

and

$$\mathrm{CI}^{ext}_\mu(\pi) = H(XY|\Pi) = H(XY) - \mathrm{IC}^{ext}_\mu(\pi).$$

With this definition it is easy to see that the information cost of a protocol π with transcript Π only depends on the distribution of μ_Π. Indeed

$$\mathrm{CI}_\mu(\pi) = H_{XY\sim\mu}(X|\Pi Y) + H_{XY\sim\mu}(Y|\Pi X) = \mathop{\mathbb{E}}_{\Pi} H_{XY\sim\mu_\Pi}(X|Y) + \mathop{\mathbb{E}}_{\Pi} H_{XY\sim\mu_\Pi}(Y|X).$$

Another nice property of concealed information is that if π_0 and π_1 are the two branches of the protocol π corresponding respectively to $B = 0$ and $B = 1$ where B is the first bit sent, then

$$\mathrm{CI}_\mu(\pi) = \Pr[B = 0]\,\mathrm{CI}_{\mu|B=0}(\pi_0) + \Pr[B = 1]\,\mathrm{CI}_{\mu|B=1}(\pi_1).$$

Thus, the expected value of CI is preserved throughout the execution of the protocol. Similar results hold for $\mathrm{CI}^{ext}_\mu(\pi)$.

3 Main Results

In this section, we state and discuss our main results in full detail. Simpler proofs are presented in this section, but the proofs of the more involved results are postponed to later sections.

We will use the following simple estimate:

$$x \in [0,1/2] \Longrightarrow x \log \frac{1}{x} \leq h(x) \leq 2x \log \frac{1}{x}, \tag{10}$$

which holds since in that range $-x \log x \geq -(1-x)\log(1-x)$.

Denote

$$\overline{h}(x) = h(\min(x, 1/2)). \tag{11}$$

It satisfies $\overline{h}(x) \geq h(x)$ and $x \leq \overline{h}(x)$. It is easy to see that h is concave. Therefore, $\overline{h}$ is also concave as it is piecewise differentiable with non increasing derivative. Additionally, $h(0) = \overline{h}(0) = 0$. We will next show how to utilize these two properties of h and $\overline{h}$: for any concave function $g\colon \mathbb{R}^+ \to \mathbb{R}$ for which $g(0) = 0$, and for any $x > 0$ and $0 < q < 1$, it holds that

$$g(qx) \geq qg(x) + (1-q)g(0) = qg(x). \tag{12}$$

This implies the subadditivity of g: for all $a_1, a_2 > 0$, $g(a_1 + a_2) \leq g(a_1) + g(a_2)$, as $g(a_i) \geq \frac{a_i}{a_1+a_2} g(a_1 + a_2)$, for all $i = 1, 2$.

3.1 Information complexity with point-wise error

Consider a communication problem $f\colon \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$, and a distribution μ. How close can $\mathrm{IC}_\mu(f, \epsilon)$ be to $\mathrm{IC}_\mu(f, 0)$? A simple argument shows that $\mathrm{IC}_\mu(f, \epsilon) \leq \mathrm{IC}_\mu(f, 0) - \Omega(\epsilon)$.

▶ **Proposition 4.** *Let $f\colon \mathcal{X}\times\mathcal{Y} \to \mathcal{Z}$, and let μ be a measure on $\mathcal{X}\times\mathcal{Y}$. Denoting $c = \mathrm{IC}_\mu(f, 0)$, we have*

$$\mathrm{IC}_\mu(f, \epsilon) \leq (1-\epsilon)\,\mathrm{IC}_\mu(f, 0) = \mathrm{IC}_\mu(f, 0) - c\epsilon.$$

Proof. Let π be a zero-error protocol for f. Consider a protocol π' in which Alice and Bob use their public randomness to run with probability $1 - \epsilon$ the protocol π, or to terminate with an arbitrary output with probability ϵ. Let Π and Π' be respectively the transcripts of π and π' on the random input (X, Y). We have

$$I(X;\Pi'|Y) = H(X|Y) - H(X|\Pi'Y) = H(X|Y) - \epsilon H(X|Y) - (1-\epsilon)H(X|\Pi Y) = (1-\epsilon)I(X;\Pi|Y).$$

The same holds for $I(Y;\Pi'|X)$, and the statement follows. ◀

Our first major theorem shows that this trivial bound can be improved to $\mathrm{IC}_\mu(f, \epsilon) \leq \mathrm{IC}_\mu(f, 0) - \Omega(h(\epsilon))$.

▶ **Theorem 5.** *Consider a function $f\colon \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$ and a probability measure μ on $\mathcal{X} \times \mathcal{Y}$ such that $\mathrm{IC}_\mu(f, 0) > 0$. There exist positive constants τ, ϵ_0, depending on f and μ (and thus on $|\mathcal{X}|, |\mathcal{Y}|, |\mathcal{Z}|$), such that for every $\epsilon \leq \epsilon_0$,*

$$\mathrm{IC}_\mu(f, \epsilon) \leq \mathrm{IC}_\mu(f, 0) - \tau h(\epsilon).$$

Moreover:

Non-constant case: *Suppose that $f(a) \neq f(b)$ for two points a, b in the support of μ, and on the same row or column. Then one can take $\tau = \mu(a)^2\mu(b)/32$, and ϵ_0 depends only on $\min(\mu(a), \mu(b))$ and $|\mathcal{X} \times \mathcal{Y}|$.*

AND case: *Let $x_0, x_1 \in \mathcal{X}$ and $y_0, y_1 \in \mathcal{Y}$. Suppose that $f(x_0y_0) = f(x_0y_1) = f(x_1y_0) = z_0$ and $f(x_1y_1) = z_1 \neq z_0$, and that $x_0y_0, x_0y_1, x_1y_0 \in \operatorname{supp}\mu$. Then one can take $\tau = \frac{\mu(x_0y_0)^2}{64}\min(\mu(x_0y_1), \mu(x_1y_0))$, and ϵ_0 depends only on $|\mathcal{X} \times \mathcal{Y}|$ and the minimum of $\mu(x_0y_0)$, $\mu(x_0y_1)$, $\mu(x_1y_0)$.*

Proof. See Section 4.1.1. ◀

▶ Remark. We prove Theorem 5 by taking a zero-error protocol for f, and turning it into an ϵ-error protocol that has an $\Omega(h(\epsilon))$ gain in the information cost over the original protocol. The high-level idea is that one of the players checks her/his input and if it is equal to a certain value x_1, then with probability ϵ changes to a different value x_0. This obviously creates an error of at most ϵ. In the Non-constant case of Theorem 5, the points a and b are used to determine x_0 and x_1, and in the AND case, the same x_0 and x_1 as they are described in the statement of the theorem can be used. Note that this modification can only create errors that erroneously output $f(x_0, y)$ instead of $f(x_1, y)$ for some values of y. This allows us to obtain a one-sided error for many functions. We shall use this later in Corollary 11 to obtain an upper bound on the information complexity of the AND function when only one-sided error is allowed.

Despite the simplicity of the idea described in Remark 3.1, the proof is rather involved, and uses some of our other results such as characterization of internal-trivial measures. The heart of the proof is of course showing the existence of appropriate values of x_0 and x_1 that can lead to the desired gain of $\Omega(h(\epsilon))$.

Let XOR denote the 2-bit XOR function. The next result shows that the analogue of Theorem 5 does not hold for the external information complexity.

▶ **Proposition 6.** *Let μ be the distribution defined as*

$$\mu = \begin{array}{|c|c|}\hline 1/2 & 0 \\ \hline 0 & 1/2 \\ \hline \end{array}.$$

Then $\mathrm{IC}^{ext}_{\mu}(\mathrm{XOR}, \epsilon) \geq \mathrm{IC}^{ext}_{\mu}(\mathrm{XOR}, 0) - 3\epsilon$.

Proof. See Section 4.1.3. ◀

For the lower bound we prove the following theorem.

▶ **Theorem 7.** *For all f, μ, ϵ, we have*

$$\mathrm{IC}_{\mu}(f, \epsilon) \geq \mathrm{IC}_{\mu}(f, 0) - 4|\mathcal{X}||\mathcal{Y}|\overline{h}(\sqrt{\epsilon}).$$

Proof. See Section 4.1.2. ◀

Theorem 7 is obtained by taking an ϵ-error protocol and completing it to a zero-error protocol. Here Alice and Bob first run the protocol that performs $[f, \epsilon]$, but when this protocol terminates, instead of returning the output, they continue their interaction to verify that the value that they have obtained is correct. We will be able to show that these additional interactions can be performed at a small information cost, and thus the total information complexity of the new protocol is not going to be much larger than that of the original protocol. This method, that we call *protocol completion*, is used in the proofs of other results such as Theorem 9 as well.

Finally let us remark that we do not know whether the bound in Theorem 7 is tight. In fact we are not aware of any examples of f and μ that refutes the possibility that $\mathrm{IC}_{\mu}(f, \epsilon) = \mathrm{IC}_{\mu}(f, 0) - \Theta(h(\epsilon))$ for every f and μ satisfying $\mathrm{IC}_{\mu}(f, 0) > 0$.

3.2 Information complexity with distributional error

In Section 3.1 we considered the amount of gain one can obtain by allowing point-wise error. Next we turn to distributional error. How much can one gain in information cost by allowing a distributional error of ϵ? Small modifications in the proofs of Theorem 5 and Theorem 7 imply the following bounds.

▶ **Theorem 8.** *Let μ be a probability measure on $\mathcal{X} \times \mathcal{Y}$, and let $f\colon \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$ satisfy $\mathrm{IC}_\mu(f, \mu, 0) > 0$. We have*

$$\mathrm{IC}_\mu(f,\mu,0) - 4|\mathcal{X}||\mathcal{Y}|\overline{h}(\sqrt{\epsilon/\alpha}) \leq \mathrm{IC}_\mu(f,\mu,\epsilon) \leq \mathrm{IC}_\mu(f,\mu,0) - \frac{\alpha^2}{4} h\left(\epsilon\alpha/4\right) + 3\epsilon \log|\mathcal{X} \times \mathcal{Y}|,$$

where $\alpha = \min_{xy \in \operatorname{supp} \mu} \mu(x,y)$.

Proof. See Section 4.2. ◀

It is also possible to prove the upper bound of Theorem 8 using a different approach by "truncating" a zero-error protocol. Unfortunately this approach requires some assumptions on the support of μ. Nevertheless we sketch this proof, as the idea seems to be new, and it might have other applications.

Let $\Delta_0 \subseteq \Delta(\mathcal{X} \times \mathcal{Y})$ be the set of all measures ν such that $\mathrm{IC}_\nu(f, \nu, \epsilon) = 0$. Consider a protocol π that performs $[f, \mu, 0]$. First we simulate π with another protocol π' such that no signal of π' jumps from outside of Δ_0 to the interior of Δ_0. In other words if some partial transcript t satisfies $\mu_t \notin \Delta_0$, then when the next signal B is sent, μ_{tB} is either still outside of Δ_0 or it is on the boundary $\partial\Delta_0$. The simulation can be done in a perfect manner so that if Π and Π' denote, respectively, the transcripts of π and π', then $\mu_{\Pi'}$ has the same distribution as μ_Π. The new protocol π' might not necessarily have bounded communication, but it will terminate with probability 1. We refer the reader to [14, Signal Simulation Lemma] and [4, Claim 7.14] for more details on such simulations.

We will truncate π' in the following manner to obtain a new protocol π_0 that performs $[f, \mu, \epsilon]$. Whenever the corresponding random walk of π' reaches a distribution ν that is on the boundary $\partial\Delta_0$, the two players stop the random walk, and use $\mathrm{IC}_\nu(f, \nu, \epsilon) = 0$ to output a value that creates a distributional error of at most ϵ with respect to ν at no information cost. Obviously the distributional error of the protocol π_0 is at most ϵ. To analyze its information cost, denote the transcript of π_0 by P, and note that P is a partial transcript for π'. Let π'_P be the continuation of π' when one starts at this partial transcript. It is not difficult to see that

$$\mathrm{IC}_\mu(\pi) = \mathrm{IC}_\mu(\pi') = \mathrm{IC}_\mu(\pi_0) + \mathop{\mathbb{E}}_P[\mathrm{IC}_{\mu_P}(\pi'_P)].$$

Since π' performs $[f, \mu, 0]$, the tail protocol π_P must perform $[f, \mu_P, 0]$. Hence in order to finish the proof, it suffices to show that $\mathrm{IC}_\nu(f, \nu, 0) = \Omega(h(\epsilon))$ for every $\nu \in \partial\Delta_0$, as this would imply the desired $\mathrm{IC}_\mu(\pi) \geq \mathrm{IC}_\mu(\pi_0) + \Omega(h(\epsilon))$. This can be proven with some work when μ is of full support, however it is not true for general measures. For example, consider the AND function, and let μ be the distribution on $\{0,1\}^2$ defined as $\mu(0,0) = 1 - 2\epsilon$ and $\mu(1,0) = \mu(1,1) = \epsilon$. Note that although μ is on the boundary of Δ_0, we have $\mathrm{IC}_\mu(\mathrm{AND}, \mu, 0) \leq 2\epsilon$. Indeed, since $\mu(0,1) = 0$, Bob with probability 1 knows the correct output by looking at his own input Y, and so if he sends his bit to Alice, they will both know the correct output. This will have information cost at most $H(Y|X) = \Pr[X = 1]H(Y|X = 1) = 2\epsilon$.

3.3 Information complexity of the AND function with error

Building upon the previous works of Ma and Ishwar [23, 24], Braverman et al. [4] developed a method for proving the optimality of information complexity and applied it to determine the internal and external information complexity of the two-bit AND function. They introduced a "continuous-time" protocol for this task, and proved that it has optimal internal and external information cost for any underlying distribution. Although this protocol is not a conventional communication protocol as it has access to a continuous clock, it can be approximated by conventional communication protocols through dividing the time into finitely many discrete units. Then in [4, Problem 1.1] they considered the case where error is allowed, and conjectured a gain of $\mathrm{IC}(\mathrm{AND}) - \mathrm{IC}(\mathrm{AND}, \epsilon) = \Theta(h(\epsilon))$. In this section, we conduct a thorough analysis of the information complexity of the AND function when error is permitted, and among other results, prove the aforementioned conjecture.

Applying our general bounds from in Section 3.1 and Section 3.2 (i.e. Theorems 5, 7, and 8) we already obtain that for small enough $\epsilon \geq 0$,

(i) For every distribution μ satisfying $\mathrm{IC}_\mu(\mathrm{AND}, 0) > 0$, we have

$$\mathrm{IC}_\mu(\mathrm{AND}, 0) - O_\mu(h(\sqrt{\epsilon})) \leq \mathrm{IC}_\mu(\mathrm{AND}, \epsilon) \leq \mathrm{IC}_\mu(\mathrm{AND}, 0) - \Omega_\mu(h(\epsilon));$$

(ii) For every distribution μ satisfying $\mathrm{IC}_\mu(\mathrm{AND}, \mu, 0) > 0$, we have

$$\mathrm{IC}_\mu(\mathrm{AND}, \mu, 0) - O_\mu(h(\sqrt{\epsilon})) \leq \mathrm{IC}_\mu(\mathrm{AND}, \mu, \epsilon) \leq \mathrm{IC}_\mu(\mathrm{AND}, \mu, 0) - \Omega_\mu(h(\epsilon)).$$

We show that under some conditions on the support of μ, the above lower bounds can be improved to match the upper bounds.

▶ **Theorem 9.** *For small enough $\epsilon \geq 0$, the following hold,*

(i) *For every distribution μ which is full support, except perhaps for $\mu(1,1)$, we have*

$$\mathrm{IC}_\mu(\mathrm{AND}, \epsilon) = \mathrm{IC}_\mu(\mathrm{AND}, 0) - \Theta(\overline{h}(\epsilon)),$$

where the hidden constants can be fixed if $\mu(0,0), \mu(0,1), \mu(1,0)$ are bounded away from 0.

(ii) *In particular for every distribution μ of full support, we have*

$$\mathrm{IC}_\mu(\mathrm{AND}, \mu, \epsilon) = \mathrm{IC}_\mu(\mathrm{AND}, \mu, 0) - \Theta(\overline{h}(\epsilon)).$$

Note that for every distribution μ of full support, we have $\mathrm{IC}_\mu(\mathrm{AND}, \mu, 0) = \mathrm{IC}_\mu(\mathrm{AND}, 0) > 0$, and $\mathrm{IC}_\mu(\mathrm{AND}, \epsilon/\alpha) \leq \mathrm{IC}_\mu(\mathrm{AND}, \mu, \epsilon) \leq \mathrm{IC}_\mu(\mathrm{AND}, \epsilon)$ where $\alpha = \min_{xy} \mu(xy)$. Thus Theorem 9(ii) follows from (i).

From a technical point of view, Theorem 9 is perhaps our most involved result in this article, and its proof occupies the bulk of Section 6. The first idea that facilitates the proof substantially is developed by the first two authors in [12]. They showed that it is possible to parametrize the space of the distributions $\Delta(\mathcal{X} \times \mathcal{Y})$ so that the changes that occur in the prior distribution by the players' interactions can be captured by product measures. This idea, that is discussed in details in Section 5, allows us to first prove the lower bound of Theorem 9 for the product measures, and then add minor adjustments to adopt it for non-product distributions. The second component of the proof is a stability result. Recall from Section 2.5 that the information cost of every protocol π depends only on its "leaf distribution", i.e. the distribution of μ_Π, where Π is the transcript of π or equivalently $\mu_{\boldsymbol{\ell}}$ where $\boldsymbol{\ell}$ is a random leaf of the protocol tree. Our stability result, Theorem 30, shows that the leaf distribution of every almost optimal protocol π for $[\mathrm{AND}, 0]$ shares certain similarities with that of the buzzer protocol. Note that since π does not make any errors, by the end of

the protocol, either both players know that the input is $(1,1)$, or one of them has revealed that her input is 0. Theorem 30 formalizes the intuition that in this latter case, the other player must not have revealed that his input is very likely to be 0. This is achieved through defining a potential function that depends only on the distribution of μ_Π and proving that it is bounded by the so called information wastage $\mathrm{IC}_\mu(\pi) - \mathrm{IC}_\mu(\mathrm{AND}, 0)$. With these results in hand, in order to complete the lower bound of Theorem 9, we start with a protocol π performing $[\mathrm{AND}, \epsilon]$ with almost optimal information complexity. First we show that π can be completed to a protocol that performs $[\mathrm{AND}, 0]$ at a small additional information cost, though possibly larger than the desired $O(h(\epsilon))$. Then we apply the stability result to deduce certain properties for the leaf distribution of π. This will imply that one indeed needs only an additional cost of $O(h(\epsilon))$ to extend π to a protocol that solves $[\mathrm{AND}, 0]$.

Braverman et al. [4] showed that $\mathrm{IC}(\mathrm{AND}, 0) = \max_\mu \mathrm{IC}_\mu(\mathrm{AND}, 0)$ is attained on a distribution having full support. This enables us to derive the following corollary on prior-free information complexity.

▶ **Corollary 10.** *When $\epsilon \geq 0$ is sufficiently small, we have*
(i) $\mathrm{IC}(\mathrm{AND}, \epsilon) = \mathrm{IC}(\mathrm{AND}, 0) - \Theta(\overline{h}(\epsilon))$*;*
(ii) $\mathrm{IC}^D(\mathrm{AND}, \epsilon) = \mathrm{IC}(\mathrm{AND}, 0) - \Theta(\overline{h}(\epsilon))$*;*

Proof. The measure μ that maximizes $\mathrm{IC}_\mu(\mathrm{AND}, 0)$ has full support [4], and thus $\mathrm{IC}(\mathrm{AND}, 0) = \mathrm{IC}_\mu(\mathrm{AND}, 0) = \mathrm{IC}_\mu(\mathrm{AND}, \mu, 0)$. By Theorem 9(ii),

$$\begin{aligned}\mathrm{IC}(\mathrm{AND}, \epsilon) &\geq \mathrm{IC}^D(\mathrm{AND}, \epsilon) \geq \mathrm{IC}_\mu(\mathrm{AND}, \mu, \epsilon) \geq \mathrm{IC}_\mu(\mathrm{AND}, \mu, 0) - O(\overline{h}(\epsilon)) \\ &= \mathrm{IC}(\mathrm{AND}, 0) - O(\overline{h}(\epsilon)).\end{aligned}$$

Moreover by a general upper bound that we prove later in Theorem 17, we have

$$\mathrm{IC}^D(\mathrm{AND}, \epsilon) \leq \mathrm{IC}(\mathrm{AND}, \epsilon) \leq \mathrm{IC}(\mathrm{AND}, 0) - \Omega(\overline{h}(\epsilon)).$$

Both items in the corollary follow. ◀

Since the difficult distributions for the set disjointness function are the ones in which the inputs typically have small or no intersections at all, the distributions for the AND function that assign a very small or 0 mass to the point $(1,1)$ are of particular importance. Let

$$\mathrm{IC}^\delta(\mathrm{AND}, \epsilon, 1 \to 0) = \sup_{\mu\colon \mu(1,1) \leq \delta} \mathrm{IC}_\mu(\mathrm{AND}, \epsilon, 1 \to 0).$$

The following corollary is used in Section 3.4 to analyze the information complexity of the set disjointness problem.

▶ **Corollary 11.** *When $\epsilon \geq 0$ is sufficiently small, we have*
(i) $\mathrm{IC}^0(\mathrm{AND}, \epsilon) = \mathrm{IC}^0(\mathrm{AND}, 0) - \Theta(\overline{h}(\epsilon))$*;*
(ii) $\mathrm{IC}^0(\mathrm{AND}, \epsilon, 1 \to 0) = \mathrm{IC}^0(\mathrm{AND}, 0) - \Theta(\overline{h}(\epsilon))$.
(iii) *There exist universal constants C_1 and C_2 such that for every $\epsilon, \delta > 0$,*

$$\mathrm{IC}^\delta(\mathrm{AND}, \epsilon, 1 \to 0) \leq \mathrm{IC}^0(\mathrm{AND}, 0) - C_1\overline{h}(\epsilon) + C_2\overline{h}(\delta).$$

Proof. Let μ be the distribution maximizing $\mathrm{IC}_\mu(\mathrm{AND}, 0)$ under the constraint $\mu(1,1) = 0$; This measure, which is described in [4], has full support except for $\mu(1,1) = 0$. Thus by Theorem 9(i),

$$\mathrm{IC}^0(\mathrm{AND}, \epsilon) \geq \mathrm{IC}_\mu(\mathrm{AND}, \epsilon) \geq \mathrm{IC}_\mu(\mathrm{AND}, 0) - O(\overline{h}(\epsilon)) = \mathrm{IC}^0(\mathrm{AND}, 0) - O(h(\epsilon)).$$

Consequently, since $\mathrm{IC}^0(\mathrm{AND},\epsilon) \le \mathrm{IC}^0(\mathrm{AND},\epsilon,1\to 0)$, both (i) and (ii) will follow if we prove $\mathrm{IC}^0(\mathrm{AND},\epsilon,1\to 0) \le \mathrm{IC}^0(\mathrm{AND},0) - \Omega(\overline{h}(\epsilon))$. To prove this, we would like to apply the AND case of Theorem 5, however to be able to obtain a uniform upper bound on $\mathrm{IC}^0(\mathrm{AND},\epsilon,1\to 0)$, we need to have a uniform lower bound on the probabilities $\mu(0,0),\mu(0,1),\mu(1,0)$. Let $\alpha > 0$ to be determined later, and consider any distribution μ with $\mu(1,1) = 0$ and $\mu(a) < \alpha$ for some input $a \neq (1,1)$. Pick $b \in \{0,1\}^2 \setminus \{a,(1,1)\}$, and obtain the distribution μ' from μ by transferring all the probability mass on a to b. That is $\mu'(b) = \mu(a) + \mu(b)$ and $\mu'(a) = 0$, and otherwise μ and μ' are identical. Obviously $|\mu - \mu'| = \alpha$. Now (3) and (12) imply

$$\mathrm{IC}_\mu(\mathrm{AND},\epsilon,1\to 0) \le \mathrm{IC}_\mu(\mathrm{AND},0) \le \mathrm{IC}_{\mu'}(\mathrm{AND},0)+4\alpha+2h(2\alpha) = 4\alpha+2h(2\alpha) \le 4h(2\alpha), \tag{13}$$

where we used the fact that $\mathrm{IC}_{\mu'}(\mathrm{AND},0) = 0$ as $\operatorname{supp}\mu'$ contains only two points. Setting $\alpha = 0.001$ for example yields $\mathrm{IC}_\mu(\mathrm{AND},0) \le 4h(2\alpha) < 0.1 < \mathrm{IC}^0(\mathrm{AND},0) \approx 0.4827$. It remains to prove the statement for the distributions μ with $\mu(0,0),\mu(0,1),\mu(1,0) \ge \alpha$. In this case Theorem 5 (See Remark 3.1 regarding the one-sidedness) implies that exists a constant $C > 0$ such that $\mathrm{IC}_\mu(\mathrm{AND},\epsilon,1\to 0) \le \mathrm{IC}^0(\mathrm{AND},0) - C\overline{h}(\epsilon)$. This finishes the proof (i) and (ii).

To prove (iii), consider an arbitrary distribution μ with $\mu(1,1) \le \delta$, and let μ' be the distribution that is obtained from μ by moving the probability mass on $(1,1)$ to a different point so that $\mu'(1,1) = 0$ and $|\mu - \mu'| = \delta$. Similar to (13), we obtain

$$\mathrm{IC}_\mu(\mathrm{AND},\epsilon,1\to 0) \le \mathrm{IC}_{\mu'}(\mathrm{AND},\epsilon,1\to 0) + 4h(2\delta) \le \mathrm{IC}^0(\mathrm{AND},\epsilon,1\to 0) + 4h(2\delta),$$

and thus (iii) follows from (ii). ◀

3.4 Set disjointness function with error

In this section we focus on the set disjointness function. Firstly it is not hard to obtain the following result.

▶ **Corollary 12.** *For $\epsilon \ge 0$ small enough,*

$$\mathrm{IC}(\mathrm{DISJ}_n,\epsilon) \ge n[\mathrm{IC}^0(\mathrm{AND},0) - \Theta(h(\epsilon))],$$

where the hidden constant is independent of n.

Proof. By the argument that proves the additivity of information complexity (see e.g. [6]), one can prove that $\mathrm{IC}(\mathrm{DISJ}_n,\epsilon) \ge n\,\mathrm{IC}^0(\mathrm{AND},\epsilon)$. Then apply Corollary 10. The essential idea is the following. Consider a distribution μ on $\{0,1\}^2$ with $\mu(1,1) = 0$, and let $(a,b) \in \{0,1\}^2$ be an input for the AND function. Let $XY \in \{0,1\}^n \times \{0,1\}^n$ be such that for some randomly selected $J \in \{1,\dots,n\}$ we have $(X_j,Y_j) = (a,b)$, and for $i \in \{1,\dots,n\} \setminus \{J\}$, the pairs (X_i,Y_i) are i.i.d. random variables, each with distribution μ. Since $\mu(1,1) = 0$, we have $\mathrm{DISJ}_n(X,Y) = 1 - \mathrm{AND}(a,b)$ with probability 1. Thus one can take a protocol π for DISJ_n and use it to solve $\mathrm{AND}(a,b)$ correctly for every (a,b). By sampling XY in a clever way, using both public and private randomness, one can guarantee that the information cost of the new protocol that solves $\mathrm{AND}(a,b)$ will be the information cost of π divided by n. ◀

As a result one also obtains that $R_\epsilon(\mathrm{DISJ}_n) \ge n[\mathrm{IC}^0(\mathrm{AND},0) - \Theta(h(\epsilon))]$. It turns out that by using techniques from [4] and [3], one can prove the following theorem.

▶ **Theorem 13.** *For the set disjointness function* DISJ_n *on inputs of length* n*, we have*

$$R_\epsilon(\mathrm{DISJ}_n) = n[\mathrm{IC}^0(\mathrm{AND}, 0) - \Theta(h(\epsilon))].$$

Proof. See Section 7.1. ◀

We conjecture that in fact the exact constant is given by $\mathrm{IC}^0(\mathrm{AND}, \epsilon, 1 \to 0)$. In other words:

▶ **Conjecture 14.** *For the set disjointness function* DISJ_n *on inputs of length* n*, we have*

$$R_\epsilon(\mathrm{DISJ}_n) = n\,\mathrm{IC}^0(\mathrm{AND}, \epsilon, 1 \to 0) + o(n).$$

Braverman [3] proved that for all $0 < \alpha < 1$ and for all functions f,

$$\mathrm{IC}^D(f, \epsilon) \geq (1 - \alpha)\,\mathrm{IC}(f, \frac{\epsilon}{\alpha}).$$

When $f = \mathrm{DISJ}_n$, Corollary 12 gives

$$\frac{\mathrm{IC}^D(\mathrm{DISJ}_n, \epsilon)}{n} \geq (1 - \alpha)(\mathrm{IC}^0(\mathrm{AND}, 0) - \Theta(h(\epsilon/\alpha))) \geq \mathrm{IC}^0(\mathrm{AND}, 0) - \Theta(\alpha + h(\epsilon/\alpha)).$$

Substituting $\alpha = \sqrt{\epsilon \log(1/\epsilon)}$ yields

$$\frac{\mathrm{IC}^D(\mathrm{DISJ}_n, \epsilon)}{n} \geq \mathrm{IC}^0(\mathrm{AND}, 0) - \Theta(\sqrt{h(\epsilon)}). \tag{14}$$

In Theorem 15 below, which is one of our main contributions, we show that this bound is sharp. The proof relies on introducing a new protocol for set disjointness problem, and analyzing its information cost.

▶ **Theorem 15.** *For the set disjointness function* DISJ_n *on inputs of length* n*, we have*

$$\mathrm{IC}^D(\mathrm{DISJ}_n, \epsilon) = n[\mathrm{IC}^0(\mathrm{AND}, 0) - \Theta(\sqrt{h(\epsilon)})] + O(\log n).$$

Proof. See Section 7.2. ◀

3.5 Prior-free Information Cost

Theorem 15 shows that for $\alpha = \sqrt{\epsilon \log(1/\epsilon)} = \Theta(\sqrt{h(\epsilon)})$, and sufficiently large n, we have

$$\frac{\mathrm{IC}^D(\mathrm{DISJ}_n, \epsilon)}{1 - \Theta(\alpha)} = \mathrm{IC}(\mathrm{DISJ}_n, \epsilon/\alpha) < \mathrm{IC}(\mathrm{DISJ}_n, \epsilon),$$

and thus proves a separation between distributional and non-distributional prior-free information complexity. As we discussed in the introduction this has the important implication that amortized randomized communication complexity is not necessarily equal to the amortized distributional communication complexity with respect to the hardest distribution. More precisely, there are examples for which $\max_\mu D^{\mu,n}_\epsilon(f^n) \neq R^n_\epsilon(f^n)$.

Next we turn to proving general lower bounds and upper bounds for the prior-free information complexity. Theorem 7 immediately implies a lower bound for non-distributional prior-free information complexity.

▶ **Corollary 16** (corollary of Theorem 7)**.** *For every function* f *and* $0 \leq \epsilon \leq 1$*, we have*

$$\mathrm{IC}(f, \epsilon) \geq \mathrm{IC}(f, 0) - 4|\mathcal{X} \times \mathcal{Y}|\overline{h}(\sqrt{\epsilon}).$$

Since unless μ satisfies certain conditions, Theorem 5 does not provide an upper bound on $\mathrm{IC}_\mu(f,\epsilon)$ that is uniform on μ, we cannot apply it directly to bound $\mathrm{IC}(f,\epsilon)$. However, we will get around this problem by proving that the "difficult distributions" satisfy these conditions and hence we obtain the desired upper bound.

▶ **Theorem 17.** *If $f\colon \mathcal{X}\times\mathcal{Y}\to\mathcal{Z}$ is non-constant, then*

$$\mathrm{IC}(f,\epsilon) \le \mathrm{IC}(f,0) - \Omega(h(\epsilon)),$$

where the hidden constant depends on f.

Proof. See Section 4.3. ◀

The same upper bound and lower bound hold for $\mathrm{IC}^D(f,\epsilon)$.

▶ **Theorem 18.** *If $f\colon \mathcal{X}\times\mathcal{Y}\to\mathcal{Z}$ is non-constant, then*

$$\mathrm{IC}^D(f,0) - O(\overline{h}(\sqrt{\epsilon})) \le \mathrm{IC}^D(f,\epsilon) \le \mathrm{IC}^D(f,0) - \Omega(h(\epsilon)),$$

where the hidden constants depend on f.

Proof. It is shown in [3] that $\mathrm{IC}^D(f,0) = \mathrm{IC}(f,0)$, and thus the upper bound follows from Theorem 17 as $\mathrm{IC}^D(f,\epsilon) \le \mathrm{IC}(f,\epsilon)$.

To prove the lower bound, choose a measure μ that maximizes $\mathrm{IC}_\mu(f,\mu,0)$, and let $\alpha = \min_{xy\in\operatorname{supp}\mu}\mu(x,y)$. Applying Theorem 8, we get

$$\mathrm{IC}^D(f,\epsilon) \ge \mathrm{IC}_\mu(f,\mu,\epsilon) \ge \mathrm{IC}_\mu(f,\mu,0) - 4|\mathcal{X}||\mathcal{Y}|\overline{h}(\sqrt{\epsilon/\alpha}) = \mathrm{IC}^D(f,0) - O(\overline{h}(\sqrt{\epsilon})).$$ ◀

3.6 A characterization of trivial measures

We start with a few of definitions. Let $f\colon \mathcal{X}\times\mathcal{Y}\to\mathcal{Z}$ be an arbitrary function, and μ a distribution on $\mathcal{X}\times\mathcal{Y}$. We say that μ is *external-trivial* if $\mathrm{IC}^{ext}_\mu(f,0) = 0$. We say that μ is *strongly external-trivial* if there exists a protocol π computing f correctly on all inputs satisfying $\mathrm{IC}^{ext}_\mu(\pi) = 0$. We say that μ is *structurally external-trivial* if f is constant on $S_A\times S_B$, where S_A is the support of the marginal of μ on Alice's input and S_B is the support of the marginal of μ on Bob's input.

Similarly we say that μ is *internal-trivial* if $\mathrm{IC}_\mu(f,0) = 0$. We say that μ is *strongly internal-trivial* if there exists a protocol π computing f correctly on all inputs satisfying $\mathrm{IC}_\mu(\pi) = 0$. We say that μ is *structurally internal-trivial* if the marginals of μ can be partitioned as $S_A = \bigcup_i \mathcal{X}_i$ and $S_B = \bigcup_i \mathcal{Y}_i$ so that the support of μ is contained in $\bigcup_i \mathcal{X}_i\times\mathcal{Y}_i$, and f is constant on each $\mathcal{X}_i\times\mathcal{Y}_i$.

Theorem 19 below shows that all our definitions of internal triviality are equivalent. In particular, if $\mathrm{IC}_\mu(f,0) = 0$, then the infimum in the definition of IC_μ is achieved by a finite protocol.

▶ **Theorem 19.** *Let $f\colon \mathcal{X}\times\mathcal{Y}\to\mathcal{Z}$ be an arbitrary function, and μ a distribution on $\mathcal{X}\times\mathcal{Y}$.*

The distribution μ is internal-trivial iff it is strongly internal-trivial iff it is structurally internal-trivial.

Proof. See Section 4.4. ◀

In order to prove Theorem 19, we first obtain a characterization of measures that are not structurally internal-trivial, by defining a graph G_μ on the support of every distribution μ on $\mathcal{X}\times\mathcal{Y}$.

▶ **Definition 20.** Let G be the graph whose vertex set is $\mathcal{X} \times \mathcal{Y}$, and two vertices are connected if they agree on one of their coordinates. That is, $(x, y), (x, y')$ are connected for every $x \in \mathcal{X}$ and $y \neq y' \in \mathcal{Y}$, and $(x, y), (x', y)$ are connected for every $x \neq x' \in \mathcal{X}$ and $y \in \mathcal{Y}$. In short, G is the Cartesian product of the complete graphs $K_{\mathcal{X}}$ and $K_{\mathcal{Y}}$. Let G_μ be the subgraph of G induced by the support of μ. For every connected component C of G_μ, define

$$C_A = \{x \in \mathcal{X} : xy \in C \text{ for some } y \in \mathcal{Y}\},$$
$$C_B = \{y \in \mathcal{Y} : xy \in C \text{ for some } x \in \mathcal{X}\}.$$

The following lemma shows that if μ is not structurally internal-trivial, then there exists a connected component C of G_μ such that f is not constant on $C_A \times C_B$. We will use this fact later in Section 4.1.1 in the proof of Theorem 5.

▶ **Lemma 21.** *Let $f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$ be an arbitrary function, and μ a distribution on $\mathcal{X} \times \mathcal{Y}$. Then the distribution μ is structurally internal-trivial iff for every connected component C of G_μ, the function f is constant on $C_A \times C_B$.*

Proof. Suppose first that μ is structurally internal-trivial. Thus there exist partitions $S_A = \bigcup_i \mathcal{X}_i$ and $S_B = \bigcup_i \mathcal{Y}_i$ such that the support of μ is contained in $\bigcup_i \mathcal{X}_i \times \mathcal{Y}_i$ and f is constant on $\mathcal{X}_i \times \mathcal{Y}_i$ on each i. Any connected component C of G_μ must lie in some $\mathcal{X}_i \times \mathcal{Y}_i$. Indeed, if (for example) $x_j y_j, x_j y_k \in C$ where $x_j \in \mathcal{X}_j$, $y_j \in \mathcal{Y}_j$, $y_k \in \mathcal{Y}_k$, then $x_j y_k \notin \bigcup_i \mathcal{X}_i \times \mathcal{Y}_i$. As $C \subseteq \mathcal{X}_i \times \mathcal{Y}_i$, we must have $C_A \times C_B \subseteq \mathcal{X}_i \times \mathcal{Y}_i$, hence f is constant on $C_A \times C_B$ for every connected component C.

Conversely, suppose that for every connected component C of G_μ, the function f is constant on $C_A \times C_B$. If C, C' are two different connected components then C_A, C'_A are disjoint: otherwise, if (say) $(x, y) \in C$ and $(x, y') \in C'$ then (x, y) is connected to (x, y') and so $C = C'$. Thus $\{C_A : C \text{ a connected component of } G_\mu\}$ partitions a subset $\mathcal{X}'$ of $\mathcal{X}$. Similarly, $\{C_B : C \text{ a connected component of } G_\mu\}$ partitions a subset $\mathcal{Y}'$ of $\mathcal{Y}$. We can obtain partitions of $\mathcal{X}$ and $\mathcal{Y}$ by adding the parts $\mathcal{X} \setminus \mathcal{X}'$ and $\mathcal{Y} \setminus \mathcal{Y}'$. These partitions serve as a witness that μ is structurally internal-trivial. ◀

Finally we note that the analogue of Theorem 19 holds for the external case as well.

▶ **Theorem 22.** *Let $f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$ be an arbitrary function, and μ a distribution on $\mathcal{X} \times \mathcal{Y}$.*

The distribution μ is external-trivial iff it is strongly external-trivial iff it is structurally external-trivial.

Proof. See Section 4.4. ◀

4 Proofs for general functions

In this section we present the proofs of the main results on general functions presented in Section 3.

4.1 Information complexity with point-wise error

4.1.1 Proof of Theorem 5

We discuss some notation before the proof. Consider a protocol π. For an input xy, let Π_{xy} denote the random variable corresponding to the transcript of π when it is executed on the

On input XY:
- Alice privately samples a Bernoulli random variable B with parameter ϵ.
- If $X = x_1$ and $B = 1$, Alice sets $X' = x_0$, otherwise she sets $X' = X$.
- The players run π on $X'Y$.

Figure 1 The protocol π' is obtained from a protocol π using $x_0, x_1 \in \mathcal{X}$.

input xy. Let Π denote the random variable for transcripts of π, whose distribution is given as

$$\Pr[\Pi = t] = \mathop{\mathbb{E}}_{xy} \Pr[\Pi_{xy} = t] = \sum_{xy} \Pr[xy] \Pr[\Pi_{xy} = t],$$

where $\Pr[\Pi_{xy} = t] = \Pr[\Pi = t | XY = xy]$. As usual we abbreviate $\Pr[xy] = \Pr[XY = xy]$, and $\Pr[x|y] = \Pr[X = x|Y = y]$, and so on.

The next lemma shows that under some conditions, if we modify a protocol π to a new protocol π' according to Figure 1, then the information cost will have a significant drop.

▶ **Lemma 23.** *Let μ be a distribution on $\mathcal{X} \times \mathcal{Y}$, and π be a protocol with input set $\mathcal{X} \times \mathcal{Y}$. Suppose there is a set $\mathcal{L}$ of transcripts of π that satisfies, for some $C_1 \in [0, 1]$,*
(1) $\Pr[\Pi \in \mathcal{L}] \geq C_1$;
and there are $x_0\overline{y}, x_1\overline{y}$, both in the support of μ, and $C_2 \in (0, 1], \delta \in [0, 1]$ with $C_2 > 2\delta$, such that for every $t \in \mathcal{L}$,
(2) $\Pr[XY = x_0\overline{y} | \Pi = t] \geq C_2$;
(3) $\Pr[XY = x_1\overline{y} | \Pi = t] \leq \delta$.
Let $K = \log |\mathcal{X} \times \mathcal{Y}|$. Then for sufficiently small $\epsilon > 0$ (depending on μ, C_2, δ), the protocol π' defined in Figure 1 satisfies

$$\mathrm{IC}_\mu(\pi') \leq \mathrm{IC}_\mu(\pi) - C_1 C_2 h\left(\frac{\epsilon}{2} \min\left\{1, C_2 \frac{\Pr[x_1\overline{y}]}{\Pr[x_0\overline{y}]}\right\}\right) + 3\epsilon K + \overline{h}(\delta/C_2).$$

Explicitly, the upper bound holds as long as $\frac{\Pr[x_1\overline{y}]}{\Pr[x_0\overline{y}]}\epsilon + (1 - \epsilon)\delta/C_2 \leq 1/2$.

Intuitively, this condition says that π has a set of transcripts $\mathcal{L}$ that happen with significant probability, and every transcript in $\mathcal{L}$ probabilistically differentiates between $x_0\overline{y}$ and $x_1\overline{y}$. In other words, if we we see a transcript in $\mathcal{L}$, then we know that the input was much more likely to be $x_0\overline{y}$ than to be $x_1\overline{y}$. One point to note here is that we require the two points $x_0\overline{y}$ and $x_1\overline{y}$ to be in the same column. By symmetry, if there are two points in the same row satisfying the same properties, then the claim of Lemma 23 also holds.

Proof. Consider the protocols π and π' as described in Figure 1. Note that $\Pi_{X'Y}$ is the transcript of π'. We shorthand $\Pi' = \Pi_{X'Y}$. The information cost of π' is given by

$$\mathrm{IC}_\mu(\pi') = I(X; \Pi'|Y) + I(Y; \Pi'|X) = H(X|Y) + H(Y|X) - H(X|\Pi'Y) - H(Y|\Pi'X),$$

while

$$\mathrm{IC}_\mu(\pi) = I(X; \Pi|Y) + I(Y; \Pi|X) = H(X|Y) + H(Y|X) - H(X|\Pi Y) - H(Y|\Pi X).$$

Hence

$$\mathrm{IC}_\mu(\pi) - \mathrm{IC}_\mu(\pi') = H(X|\Pi'Y) - H(X|\Pi Y) + H(Y|\Pi'X) - H(Y|\Pi X).$$

Note that

$$\begin{aligned} H(Y|\Pi' X) &\geq H(Y|\Pi' XB) \geq (1-\epsilon)\, H(Y|\Pi' X, (B=0)) = (1-\epsilon)\, H(Y|\Pi X) \\ &\geq H(Y|\Pi X) - \epsilon K. \end{aligned} \tag{15}$$

Similarly, for every $y \in \mathcal{Y}$ and every possible transcript t, we have

$$H(X|\Pi' Y = ty) \geq H(X|\Pi Y = ty) - \epsilon K. \tag{16}$$

We will show that for $Y = \overline{y}$ and every transcript $t \in \mathcal{L}$,

$$H(X|\Pi' Y = t\overline{y}) \geq H(X|\Pi Y = t\overline{y}) + h\left(\frac{\epsilon}{2} \min\left\{1, C_2 \frac{\Pr[x_1\overline{y}]}{\Pr[x_0\overline{y}]}\right\}\right) - \overline{h}(\delta/C_2) - \epsilon K. \tag{17}$$

Note that Condition (2) implies that for $t \in \mathcal{L}$,

$$\Pr[\Pi Y = t\overline{y}] \geq \Pr[\Pi XY = tx_0\overline{y}] = \Pr[XY = x_0\overline{y}|\Pi = t] \Pr[\Pi = t] \geq C_2 \Pr[\Pi = t].$$

Hence

$$\Pr[\Pi \in \mathcal{L}, Y = \overline{y}] \geq C_2 \Pr[\Pi \in \mathcal{L}] \geq C_1 C_2.$$

This together with (16) and (17) would show that

$$\begin{aligned} H(X|\Pi' Y) &= \sum_t \sum_{y \in \mathcal{Y}} \Pr[\Pi' Y = ty] H(X|\Pi' Y = ty) \\ &\geq \sum_t \sum_{y \in \mathcal{Y}} (1-\epsilon) \Pr[\Pi Y = ty] H(X|\Pi' Y = ty) \\ &\geq \sum_t \sum_{y \in \mathcal{Y}} \Pr[\Pi Y = ty] H(X|\Pi Y = ty) \\ &\quad + \Pr[\Pi \in \mathcal{L}, Y = \overline{y}] \left(h\left(\frac{\epsilon}{2} \min\left\{1, C_2 \frac{\Pr[x_1\overline{y}]}{\Pr[x_0\overline{y}]}\right\}\right) - \overline{h}(\delta/C_2)\right) - 2\epsilon K \\ &\geq H(X|\Pi Y) + C_1 C_2 h\left(\frac{\epsilon}{2} \min\left\{1, C_2 \frac{\Pr[x_1\overline{y}]}{\Pr[x_0\overline{y}]}\right\}\right) - 2\epsilon K - \overline{h}(\delta/C_2). \end{aligned}$$

Applying (15) would immediately give the claimed bound.

Our aim, then, is to show (17). From now on we consider exclusively $t \in \mathcal{L}$.

The idea is to consider the indicator variable $C := 1_{[X \neq x_1]}$. Since C is a deterministic function of X, we have

$$H(X|\Pi' Y = t\overline{y}) = H(XC|\Pi' Y = t\overline{y}) = H(X|C, (\Pi' Y = t\overline{y})) + H(C|\Pi' Y = t\overline{y}). \tag{18}$$

Since $\Pr[XY = x_0\overline{y}|\Pi = t] = \Pr[Y = \overline{y}|\Pi = t] \Pr[X = x_0|\Pi Y = t\overline{y}]$, by Condition (2) we obtain

$$\Pr[X = x_0|\Pi Y = t\overline{y}] \geq \Pr[XY = x_0\overline{y}|\Pi = t] \geq C_2, \tag{19}$$

and $\Pr[Y = \overline{y}|\Pi = t] \geq C_2$. Similarly, as $\Pr[XY = x_1\overline{y}|\Pi = t] = \Pr[Y = \overline{y}|\Pi = t] \Pr[X = x_1|\Pi Y = t\overline{y}]$, we obtain by Condition (3) that

$$\Pr[X = x_1|\Pi Y = t\overline{y}] = \frac{\Pr[XY = x_1\overline{y}|\Pi = t]}{\Pr[Y = \overline{y}|\Pi = t]} \leq \frac{\delta}{C_2}. \tag{20}$$

Hence using (20), the first term in (18) can be bounded as

$$\begin{aligned} H(X|C,(\Pi' Y = t\overline{y})) &\geq (1-\epsilon) H(X|C,(B\Pi' Y = 0t\overline{y})) \\ &\geq H(X|C,(\Pi Y = t\overline{y})) - \epsilon K \\ &= H(XC|\Pi Y = t\overline{y}) - H(C|\Pi Y = t\overline{y}) - \epsilon K \\ &\geq H(X|\Pi Y = t\overline{y}) - \overline{h}(\delta/C_2) - \epsilon K. \end{aligned} \tag{21}$$

To bound the second term $H(C|\Pi' Y = t\overline{y})$ in (18), we must study $\Pr[X = x_1|\Pi' Y = t\overline{y}]$. We will use

$$\Pr[C = 0|\Pi' Y = t\overline{y}] = \Pr[X = x_1|\Pi' Y = t\overline{y}] = \frac{\Pr[\Pi' XY = tx_1\overline{y}]}{\Pr[\Pi' Y = t\overline{y}]}. \tag{22}$$

Consider the numerator first. By the definition of π',

$$\begin{aligned} \Pr[\Pi' XY = tx_1\overline{y}] &= \Pr[\Pi' = t|XY = x_1\overline{y}] \Pr[x_1\overline{y}] \\ &= (\epsilon \Pr[\Pi = t|XY = x_0\overline{y}] + (1-\epsilon) \Pr[\Pi = t|XY = x_1\overline{y}]) \Pr[x_1\overline{y}] \\ &= \epsilon \Pr[\Pi XY = tx_0\overline{y}] \frac{\Pr[x_1\overline{y}]}{\Pr[x_0\overline{y}]} + (1-\epsilon) \Pr[\Pi XY = tx_1\overline{y}]. \end{aligned} \tag{23}$$

For the denominator of (22), we have

$$\Pr[\Pi' Y = t\overline{y}] \geq \Pr[\Pi' XY = tx_0\overline{y}] = \Pr[\Pi XY = tx_0\overline{y}]. \tag{24}$$

By Conditions (2) and (3),

$$\frac{\Pr[\Pi XY = tx_1\overline{y}]}{\Pr[\Pi XY = tx_0\overline{y}]} = \frac{\Pr[XY = x_1\overline{y}|\Pi = t]}{\Pr[XY = x_0\overline{y}|\Pi = t]} \leq \delta/C_2. \tag{25}$$

Combining (22), (23), (24) and (25), we obtain the following upper bound on (22):

$$\Pr[X = x_1|\Pi' Y = t\overline{y}] \leq \frac{\Pr[x_1\overline{y}]}{\Pr[x_0\overline{y}]}\epsilon + (1-\epsilon)\delta/C_2. \tag{26}$$

To obtain a lower bound for (22) note

$$\begin{aligned} \Pr[\Pi' Y = t\overline{y}] &= \sum_x \Pr[\Pi' XY = tx\overline{y}] = \sum_{x \neq x_1} \Pr[\Pi' XY = tx\overline{y}] + \Pr[\Pi' XY = tx_1\overline{y}] \\ &= \sum_{x \neq x_1} \Pr[\Pi XY = tx\overline{y}] \\ &\qquad + \epsilon \Pr[\Pi XY = tx_0\overline{y}] \frac{\Pr[x_1\overline{y}]}{\Pr[x_0\overline{y}]} + (1-\epsilon) \Pr[\Pi XY = tx_1\overline{y}] \qquad (27) \\ &\leq \sum_x \Pr[\Pi XY = tx\overline{y}] + \epsilon \Pr[\Pi XY = tx_0\overline{y}] \frac{\Pr[x_1\overline{y}]}{\Pr[x_0\overline{y}]} \\ &= \Pr[\Pi Y = t\overline{y}] + \epsilon \Pr[\Pi XY = tx_0\overline{y}] \frac{\Pr[x_1\overline{y}]}{\Pr[x_0\overline{y}]} \\ &\leq 2 \max\left\{ \Pr[\Pi Y = t\overline{y}], \Pr[\Pi XY = tx_0\overline{y}] \frac{\Pr[x_1\overline{y}]}{\Pr[x_0\overline{y}]} \right\}. \qquad (28) \end{aligned}$$

Hence by (22), (23) and (28),

$$\begin{aligned} \Pr[X = x_1|\Pi' Y = t\overline{y}] &\geq \frac{\epsilon \Pr[\Pi XY = tx_0\overline{y}] \frac{\Pr[x_1\overline{y}]}{\Pr[x_0\overline{y}]}}{2 \max\{\Pr[\Pi Y = t\overline{y}], \Pr[\Pi XY = tx_0\overline{y}] \frac{\Pr[x_1\overline{y}]}{\Pr[x_0\overline{y}]}\}} \\ &\geq \frac{\epsilon}{2} \min\left\{1, C_2 \frac{\Pr[x_1\overline{y}]}{\Pr[x_0\overline{y}]}\right\}. \end{aligned} \tag{29}$$

where we used $\Pr[\Pi XY = tx_0\overline{y}]/\Pr[\Pi Y = t\overline{y}] = \Pr[X = x_0|\Pi Y = t\overline{y}] \geq C_2$ by (19). Thus we have shown that

$$\frac{\epsilon}{2}\min\left\{1, C_2\frac{\Pr[x_1\overline{y}]}{\Pr[x_0\overline{y}]}\right\} \leq \Pr[X = x_1|\Pi' Y = t\overline{y}] \leq \frac{\Pr[x_1\overline{y}]}{\Pr[x_0\overline{y}]}\epsilon + \frac{(1-\epsilon)\delta}{C_2}. \tag{30}$$

This together with (18) and (21) gives (17) as desired, as long as $\epsilon > 0$ is small enough such that the upper bound in (30) is at most $1/2$. ◀

Theorem 5 (restated). *Consider a function $f\colon \mathcal{X}\times\mathcal{Y}\to\mathcal{Z}$ and a probability measure μ on $\mathcal{X}\times\mathcal{Y}$ such that $\mathrm{IC}_\mu(f,0) > 0$. There exist positive constants τ, ϵ_0, depending on f and μ, such that for every $\epsilon \leq \epsilon_0$,*

$$\mathrm{IC}_\mu(f,\epsilon) \leq \mathrm{IC}_\mu(f,0) - \tau h(\epsilon).$$

Moreover:

Non-constant case: *Suppose that $f(a) \neq f(b)$ for two points a, b in the support of μ, and on the same row or column. Then one can take $\tau \geq \mu(a)^2\mu(b)/32$, and ϵ_0 depends only on $\min(\mu(a),\mu(b))$ and $|\mathcal{X}\times\mathcal{Y}|$.*

AND case: *Let $x_0, x_1 \in \mathcal{X}$ and $y_0, y_1 \in \mathcal{Y}$. Suppose that $f(x_0y_0) = f(x_0y_1) = f(x_1y_0) = z_0$ and $f(x_1y_1) = z_1 \neq z_0$, and that $x_0y_0, x_0y_1, x_1y_0 \in \operatorname{supp}\mu$. Then one can take $\tau \geq \frac{\mu(x_0y_0)^2}{64}\min(\mu(x_0y_1),\mu(x_1y_0))$, and ϵ_0 depends only on $|\mathcal{X}\times\mathcal{Y}|$ and the minimum of $\mu(x_0y_0)$, $\mu(x_0y_1)$, $\mu(x_1y_0)$.*

Proof. In order to apply the assumption $\mathrm{IC}_\mu(f,0) > 0$, we will need to use our characterization of internal-trivial measures. Consider the graph G_μ defined on $\mathcal{X}\times\mathcal{Y}$ as given in Definition 20. By Theorem 19 and Lemma 21, the assumption $\mathrm{IC}_\mu(f,0) > 0$ implies the existence of a connected component C of G_μ such that f is not constant on $C_A \times C_B$. Note that $C \subseteq \operatorname{supp}\mu$, and $C_A \times C_B$ is the corresponding rectangle given by C.

Case I: f is not constant on C

As C is connected, there must be two adjacent points $a, b \in C$ such that $f(a) \neq f(b)$. By our definition of adjacency in Definition 20, without loss of generality we can assume that a, b are in the same column. Now consider any protocol π that solves $[f, 0]$. Let $\mathcal{L}_0$ be the set of the transcripts that can occur when π runs with input a; formally,

$$\mathcal{L}_0 = \{t : \Pr[\Pi_a = t] > 0\}.$$

Clearly $\Pr[\Pi \in \mathcal{L}_0] \geq \mu(a)$. As $f(a) \neq f(b)$ and π has no error, for every $t \in \mathcal{L}_0$,

$$\Pr[XY = b|\Pi = t] = 0. \tag{31}$$

Let

$$\mathcal{L} = \{t \in \mathcal{L}_0 : \Pr[XY = a|\Pi = t] \geq \mu(a)/2\}. \tag{32}$$

We claim

$$\Pr[\Pi \in \mathcal{L}] \geq \mu(a)/2. \tag{33}$$

Indeed, note

$$\sum_{t\in\mathcal{L}_0} \Pr[\Pi = t]\Pr[XY = a|\Pi = t] = \sum_t \Pr[\Pi = t]\Pr[XY = a|\Pi = t] = \mu(a),$$

use the trivial bound $\Pr[XY = a|\Pi = t] \le 1$, we have

$$\mu(a) = \sum_{t\in\mathcal{L}} \Pr[\Pi = t]\Pr[XY = a|\Pi = t] + \sum_{t\in\mathcal{L}_0\setminus\mathcal{L}} \Pr[\Pi = t]\Pr[XY = a|\Pi = t]$$
$$\le \sum_{t\in\mathcal{L}} \Pr[\Pi = t] + \frac{\mu(a)}{2}\sum_{t\in\mathcal{L}_0\setminus\mathcal{L}} \Pr[\Pi = t] = \Pr[\Pi\in\mathcal{L}] + \frac{\mu(a)}{2}(1 - \Pr[\Pi\in\mathcal{L}]),$$

which gives $\Pr[\Pi \in \mathcal{L}] \ge \mu(a)/(2-\mu(a)) \ge \mu(a)/2$, as claimed. For small enough ϵ, the set $\mathcal{L}$ and the points a, b satisfy the three conditions in Lemma 23 with $C_1 = C_2 = \mu(a)/2$ and $\delta = 0$, respectively from (33), (32) and (31). We conclude that

$$\mathrm{IC}_\mu(f,\epsilon) \le \mathrm{IC}_\mu(f,0) - \frac{\mu(a)^2}{4}h\left(\frac{\mu(b)}{4}\epsilon\right) + 3\epsilon K \text{ whenever } \frac{\mu(b)}{\mu(a)}\epsilon \le 1/2,$$

where $K = \log|\mathcal{X}\times\mathcal{Y}|$. Hence when $\epsilon \le 1/2$, by (12) we have

$$\mathrm{IC}_\mu(f,\epsilon) \le \mathrm{IC}_\mu(f,0) - \frac{\mu(a)^2\mu(b)}{16}h(\epsilon) + 3\epsilon K.$$

We can thus find $\epsilon_0 > 0$, depending only on $\mu(a), \mu(b), K$, such that for $\epsilon \le \epsilon_0$,

$$\mathrm{IC}_\mu(f,\epsilon) \le \mathrm{IC}_\mu(f,0) - \frac{\mu(a)^2\mu(b)}{32}h(\epsilon).$$

Case II: f is constant on C but not on $C_A \times C_B$

We first make a simple observation:

Property A: For any protocol π that performs $[f, 0]$, and for every transcript t of π, there exists at least one point $b \in C$ (which can depend on t) such that $\Pr[XY = b|\Pi = t] = 0$.

Indeed, otherwise f would be constant on $C_A \times C_B$ by the rectangle property of protocols (i.e. $\Pr[\Pi = t|x_1y_1]\Pr[\Pi = t|x_2y_2] = \Pr[\Pi = t|x_1y_2]\Pr[\Pi = t|x_2y_1]$ for all x_1, x_2, y_1, y_2).

Given a protocol π that performs $[f, 0]$ and a point $a \in C$, let the set $\mathcal{L}(\pi, a)$ of transcripts be defined as

$$\mathcal{L}(\pi, a) = \{t : \Pr[XY = a|\Pi = t] \ge \mu(a)/2\}.$$

The same argument as in **Case I** shows that $\Pr[\Pi \in \mathcal{L}(\pi, a)] \ge \mu(a)/2$. For any other point $b \in C$, define

$$\mathcal{L}(\pi, a, b) = \{t \in \mathcal{L}(\pi, a) : \Pr[XY = b|\Pi = t] = 0\}.$$

Let $k := |C|$; necessarily $k \ge 3$. By **Property A**, we have

$$\mathcal{L}(\pi, a) = \bigcup_{b\in C} \mathcal{L}(\pi, a, b).$$

This implies the existence of a point $b \in C$ with $\Pr[\Pi \in \mathcal{L}(\pi, a, b)] \ge \Pr[\Pi \in \mathcal{L}(\pi, a)]/k \ge \mu(a)/2k$. To sum up, we have shown that there exist two different points $a, b \in C \subseteq \operatorname{supp}\mu$ such that the set of transcripts $\mathcal{L}(\pi, a, b)$ satisfies the following properties:

(1') $\Pr[\Pi \in \mathcal{L}(\pi, a, b)] \ge \mu(a)/2k$;
(2') $\Pr[XY = a|\Pi = t] \ge \mu(a)/2$ for every $t \in \mathcal{L}(\pi, a, b)$;
(3') $\Pr[XY = b|\Pi = t] = 0$ for every $t \in \mathcal{L}(\pi, a, b)$.

Now consider a sequence of protocols π_n that all perform $[f, 0]$ and $\lim_{n\to\infty} \mathrm{IC}_\mu(\pi_n) = \mathrm{IC}_\mu(f, 0)$. Fix (arbitrarily) a point $a \in C$. For every protocol π_n we construct $\mathcal{L}(\pi_n, a, b_{\pi_n})$ as above. Since there are only $k-1$ different values of b, by picking a subsequence of π_n if necessary, without loss of generality, we may assume that for some point $b \in C$, $b_{\pi_n} = b$ for all π_n. Hence for every π_n we have a set of transcripts $\mathcal{L}(\pi_n, a, b)$ such that properties (1'), (2') and (3') are all satisfied.

If we compare these three conditions with the conditions in Lemma 23, we find that the only issue is that we do not know whether a and b are in the same row or column (in terms of the graph G_μ, whether a and b are adjacent).

Case IIa: a, b are adjacent in G_μ. As we expand on below, we can guarantee that this case happens in the *AND case* (see theorem statement) by choosing $a = x_0y_0$.

For small enough ϵ, the set $\mathcal{L}(\pi, a, b)$ and the points a, b satisfy the three conditions in Lemma 23 with $C_1 = \mu(a)/2k$, $C_2 = \mu(a)/2$ and $\delta = 0$, respectively from (1'), (2') and (3'). We conclude that

$$\mathrm{IC}_\mu(f, \epsilon) \leq \mathrm{IC}_\mu(f, 0) - \frac{\mu(a)^2}{4k} h\left(\frac{\mu(b)}{4}\epsilon\right) + 3\epsilon K \text{ whenever } \frac{\mu(b)}{\mu(a)}\epsilon \leq 1/2,$$

where $K = \log|\mathcal{X} \times \mathcal{Y}|$. Repeating the calculations of Case I, we can find $\epsilon_0 > 0$, depending only on $\mu(a), \mu(b), K$, such that for $\epsilon \leq \epsilon_0$,

$$\mathrm{IC}_\mu(f, \epsilon) \leq \mathrm{IC}_\mu(f, 0) - \frac{\mu(a)^2\mu(b)}{32k} h(\epsilon).$$

Suppose now that we are in the AND case. Choosing $a = x_0y_0$, we see that Property A must hold for some $b \in \{x_0y_1, x_1y_0\}$, since a transcript having positive probability on both x_0y_1 and x_1y_0 also has positive probability on x_1y_1, whereas $f(x_0y_1) \neq f(x_1y_1)$ by assumption. Property (1') thus holds with $k = 2$, and we conclude that for $\epsilon \leq \epsilon_0$,

$$\mathrm{IC}_\mu(f, \epsilon) \leq \mathrm{IC}_\mu(f, 0) - \frac{\mu(a)^2\mu(b)}{64} h(\epsilon).$$

Case IIb: a, b are not adjacent in G_μ. To handle this case, we run a *binary search* along a shortest path connecting a and b in C.

Pick an arbitrary point $c \in C$ in some shortest path connecting a and b. For every π_n, sort the transcripts in $\mathcal{L}(\pi_n, a, b)$ according to $p_{n,t,c} := \Pr[XY = c|\Pi_n = t]$ in increasing order, where Π_n is the random variable representing the transcript of π_n. Let m_n be the median of the sequence $p_{n,t,c}$ according to the conditional probability measure $\nu_n(t) := \Pr[\Pi_n = t | t \in \mathcal{L}(\pi_n, a, b)]$, i.e.,

$$\nu_n(\{t \in \mathcal{L}(\pi_n, a, b) : p_{n,t,c} \leq m_n\}), \nu_n(\{t \in \mathcal{L}(\pi_n, a, b) : p_{n,t,c} \geq m_n\}) \geq 1/2. \tag{34}$$

Such a median always exists: if m_n is the smallest value such that $\nu_n(\{t \in \mathcal{L}(\pi_n, a, b) : p_{n,t,c} \leq m_n\}) \geq 1/2$ then $\nu_n(\{t \in \mathcal{L}(\pi_n, a, b) : p_{n,t,c} \geq m_n\}) = 1 - \nu_n(\{t \in \mathcal{L}(\pi_n, a, b) : p_{n,t,c} < m_n\}) \geq 1/2$.

As trivially $m_n \in [0, 1]$, the sequence m_n must have a convergent subsequence. Again by picking a subsequence from m_n if necessary, we may assume that the sequence m_n itself is convergent, say $\lim_{n\to\infty} m_n = m$; moreover, if $m > 0$, by picking another subsequence we can assume that $m_n \geq m/2$ for all n. The *binary search* algorithm is then given as:

- If $m = 0$, update the set of transcripts to

$$\mathcal{L}(\pi_n, a, c) := \{t \in \mathcal{L}(\pi_n, a, b) : p_{n,t,c} \le m_n\}, \tag{35}$$

  and continue the algorithm with b replaced by c;
- If $m > 0$, update the set of transcripts to

$$\mathcal{L}(\pi_n, c, b) := \{t \in \mathcal{L}(\pi_n, a, b) : p_{n,t,c} \ge m_n\}, \tag{36}$$

  and continue the algorithm with a replaced by c.

We argue that the three properties are roughly preserved. In the case $m = 0$, Property (2') is kept, while Properties (1') and (3') change to

$$\Pr[\Pi_n \in \mathcal{L}(\pi_n, a, c)] \ge \mu(a)/4k \quad \text{and} \quad \Pr[XY = c|\Pi_n = t] \le m_n,\ \forall\, t \in \mathcal{L}(\pi_n, a, c),$$

respectively. In the case $m > 0$, Property (3') is preserved while Properties (1') and (2') change to

$$\Pr[\Pi_n \in \mathcal{L}(\pi_n, c, b)] \ge \mu(a)/4k \quad \text{and} \quad \Pr[XY = c|\Pi_n = t] > m/2,\ \forall\, t \in \mathcal{L}(\pi_n, c, b).$$

In either case, we have seen that the new set of transcripts $\mathcal{L}(\pi_n, a, b)$ together with the new two points a and b satisfy Condition (1), (2) and (3) in Lemma 23 with proper constants (e.g., δ_n in Condition (3) is at most m_n for protocol π_n, and $m_n \to 0$). After finitely many steps, the binary search algorithm has to stop and return two adjacent points a and b. Suppose that it stops after s steps; note that $s \le \lceil \log k \rceil$. Lemma 23 then gives the upper bound

$$\mathrm{IC}_\mu(f, \epsilon) \le \mathrm{IC}_\mu(\pi_n) - \frac{\mu(a)}{2^{s+1}k} C_2 h\left(\frac{\epsilon}{2}\min\{1, C_2 R\}\right) + 3\epsilon K + \overline{h}(\delta_n/C_2). \tag{37}$$

for some $C_2, R, K > 0$ (where C_2, R depend on μ) and a sequence δ_n tending to zero, assuming that

$$R\epsilon + (1-\epsilon)\delta_n/C_2 \le 1/2 \text{ and } \delta_n/C_2 \le 1/2.$$

By picking a subsequence, we can assume that $\delta_n \le C_2/4$ for all n. Lemma 23 then applies for all $\epsilon \le 1/(4R)$. Taking the limit of the right-hand side of (37) as $n \to \infty$, we obtain

$$\mathrm{IC}_\mu(f, \epsilon) \le \mathrm{IC}_\mu(f, 0) - \frac{\mu(a)}{2^{s+1}k} C_2 h\left(\frac{\epsilon}{2}\min\{1, C_2 R\}\right) + 3\epsilon K = \mathrm{IC}_\mu(f, 0) - \Omega(h(\epsilon)). \qquad \blacktriangleleft$$

4.1.2 Proof of Theorem 7

Theorem 7 (restated). *For all f, μ, ϵ, we have*

$$\mathrm{IC}_\mu(f, \epsilon) \ge \mathrm{IC}_\mu(f, 0) - 4|\mathcal{X}||\mathcal{Y}|\overline{h}(\sqrt{\epsilon}).$$

Proof of Theorem 7. Without loss of generality assume that μ is a full-support distribution as otherwise we can approximate it by a sequence of full-support distributions and appeal to the continuity of $\mathrm{IC}_\nu(f, \epsilon)$ with respect to ν. Consider a protocol π that performs $[f, \epsilon]$. For every leaf ℓ of π, let z_ℓ and μ_ℓ respectively denote the output of the leaf, and the distribution of the inputs conditioned on the leaf ℓ. We will complete it into a protocol π' that performs $[f, 0]$, as follows.

On input (X, Y):

- Alice and Bob run the protocol π and reach a leaf ℓ;
- For every $(x, y) \in \Omega_\ell := \{(x, y) : f(x, y) \neq z_\ell\}$, Alice and Bob verify whether $XY = xy$, as follows:
  - If $\mu_\ell(x) \leq \mu_\ell(y)$, Alice reveals whether $X = x$ to Bob, and if yes, Bob reveals whether $Y = y$ to Alice. If $XY = xy$, they terminate.
  - If $\mu_\ell(x) > \mu_\ell(y)$, Bob initiates the verification process.

Clearly, in the end, either both Alice and Bob already revealed their inputs to each other, or otherwise they know $XY \notin \Omega_\ell$, and hence z_ℓ is the correct output. Therefore π' performs the task $[f, 0]$.

Next we analyze $\mathrm{IC}_\mu(\pi')$. Let $\pi_{\ell,xy}$ denote the sub-protocol that starts with the distribution μ_ℓ and verifies whether $XY = xy$. In the case when Alice initiates the verification procedure, we have

$$\mathrm{IC}_{\mu_\ell}(\pi_{\ell,xy}) = h(\mu_\ell(x)) + \mu_\ell(x) h\left(\frac{\mu_\ell(x,y)}{\mu_\ell(x)}\right) \leq h(\mu_\ell(x)) + \mu_\ell(x) \leq 2\overline{h}(\mu_\ell(x)),$$

where by an abuse of notation we are denoting by $\mu_\ell(x)$ the marginal of μ_ℓ on x. We can obtain a similar bound for the case where Bob initiates the process, and hence

$$\begin{aligned}
\mathrm{IC}_{\mu_\ell}(\pi_{\ell,xy}) &\leq 2\min\{\overline{h}(\mu_\ell(x)), \overline{h}(\mu_\ell(y))\} \\
&= 2\overline{h}\Big(\mu_\ell(x,y) + \min\{\Pr_{\mu_\ell}[X \neq x, Y = y], \Pr_{\mu_\ell}[X = x, Y \neq y]\}\Big) \\
&\leq 2\overline{h}(\mu_\ell(x,y)) + 2\overline{h}\Big(\min\{\Pr_{\mu_\ell}[X \neq x, Y = y], \Pr_{\mu_\ell}[X = x, Y \neq y]\}\Big)
\end{aligned}$$

by the subadditivity of $\overline{h}$. Using the monotonicity of $\overline{h}$ together with $\min\{a, b\} \leq \sqrt{ab}$, we obtain that

$$\mathrm{IC}_{\mu_\ell}(\pi_{\ell,xy}) \leq 2\overline{h}(\mu_\ell(x,y)) + 2\overline{h}\Big(\sqrt{\Pr_{\mu_\ell}[X = x, Y \neq y] \Pr_{\mu_\ell}[X \neq x, Y = y]}\Big) \tag{38}$$

holds for every leaf ℓ and $(x, y) \in \Omega_\ell$. Let $\Pi_{\ell,xy}$ denote the transcript of $\pi_{\ell,xy}$. Since $\pi_{\ell,xy}$ is a deterministic protocol, we have $H_{\mu_\ell}(\Pi_{\ell,xy}|XY) = 0$, and thus

$$\mathrm{IC}_{\mu_\ell}(\pi_{\ell,xy}) = I(\Pi_{\ell,xy}; Y|X) + I(\Pi_{\ell,xy}; X|Y) = H_{\mu_\ell}(\Pi_{\ell,xy}|X) + H_{\mu_\ell}(\Pi_{\ell,xy}|Y).$$

Thus the sub-additivity of entropy implies that the information cost of running all the protocols $\pi_{\ell,xy}$ (for all $x, y \in \Omega_\ell$) is bounded by the sum of their individual information cost. Let ℓ be a leaf of π sampled by running π on a random input. By (38),

$$\begin{aligned}
\mathrm{IC}_\mu(\pi') - \mathrm{IC}_\mu(\pi) &\leq \mathop{\mathbb{E}}_{\ell} \sum_{xy \in \Omega_\ell} \mathrm{IC}_{\mu_\ell}(\pi_{\ell,xy}) = \sum_{(x,y) \in \mathcal{X} \times \mathcal{Y}} \mathop{\mathbb{E}}_{\ell} 1_{z_\ell \neq f(x,y)} \mathrm{IC}_{\mu_\ell}(\pi_{\ell,xy}) \\
&\leq \sum_{(x,y) \in \mathcal{X} \times \mathcal{Y}} 2 \mathop{\mathbb{E}}_{\ell} 1_{z_\ell \neq f(x,y)} \overline{h}(\mu_\ell(x,y)) + \\
&\quad \sum_{(x,y) \in \mathcal{X} \times \mathcal{Y}} 2 \mathop{\mathbb{E}}_{\ell} 1_{z_\ell \neq f(x,y)} \overline{h}\Big(\sqrt{\Pr_{\mu_\ell}[X = x, Y \neq y] \Pr_{\mu_\ell}[X \neq x, Y = y]}\Big) \\
&\leq \sum_{(x,y) \in \mathcal{X} \times \mathcal{Y}} 2\overline{h}\Big(\mathop{\mathbb{E}}_{\ell} 1_{z_\ell \neq f(x,y)} \mu_\ell(x,y)\Big) + \\
&\quad \sum_{(x,y) \in \mathcal{X} \times \mathcal{Y}} 2\overline{h}\Big(\mathop{\mathbb{E}}_{\ell} \sqrt{1_{z_\ell \neq f(x,y)} \Pr_{\mu_\ell}[X = x, Y \neq y] \Pr_{\mu_\ell}[X \neq x, Y = y]}\Big)
\end{aligned} \tag{39}$$

where we used the concavity of $\overline{h}$ in the last step.

For the first summand, we have that for every (x, y),

$$\begin{aligned}\mathop{\mathbb{E}}_{\ell} 1_{z_\ell \neq f(x,y)} \mu_\ell(x,y) &= \sum_\ell \Pr[XY = xy, \pi \text{ reaches } \ell] 1_{z_\ell \neq f(x,y)} \\ &= \sum_\ell \Pr[\pi \text{ reaches } \ell \mid XY = xy] \mu(xy) 1_{z_\ell \neq f(x,y)} \\ &= \mu(xy) \sum_\ell \Pr[\pi_{x,y} \text{ reaches } \ell] 1_{z_\ell \neq f(x,y)} = \mu(xy) \Pr[\pi(x,y) \neq f(x,y)] \\ &\leq \mu(xy)\epsilon \leq \epsilon, \end{aligned} \tag{40}$$

where we used that by definition $\mu_\ell(xy) = \Pr[XY = xy \mid \pi \text{ reaches } \ell]$, and the fact that the protocol π performs the task $[f, \epsilon]$.

For the second summand in (39), since μ_ℓ is obtained by scaling rows and columns of μ, we have

$$\frac{\Pr_\mu[X = x, Y = y] \Pr_\mu[X \neq x, Y \neq y]}{\Pr_\mu[X = x, Y \neq y] \Pr_\mu[X \neq x, Y = y]} = \frac{\Pr_{\mu_\ell}[X = x, Y = y] \Pr_{\mu_\ell}[X \neq x, Y \neq y]}{\Pr_{\mu_\ell}[X = x, Y \neq y] \Pr_{\mu_\ell}[X \neq x, Y = y]}$$

Define (recall that we assumed μ is of full support)

$$a_\ell = 1_{z_\ell \neq f(x,y)} \frac{\Pr_{\mu_\ell}[X = x, Y = y]}{\Pr_\mu[X = x, Y = y]}, \quad b_\ell = \frac{\Pr_{\mu_\ell}[X \neq x, Y \neq y]}{\Pr_\mu[X \neq x, Y \neq y]},$$

and note that

$$\begin{aligned} & 1_{z_\ell \neq f(x,y)} \Pr_{\mu_\ell}[X = x, Y \neq y] \Pr_{\mu_\ell}[Y = y, X \neq x] \\ &= a_\ell b_\ell \Pr_\mu[X = x, Y \neq y] \Pr_\mu[X \neq x, Y = y] \leq a_\ell b_\ell. \end{aligned} \tag{41}$$

Since

$$\mathop{\mathbb{E}}_{\ell} a_\ell = \frac{1}{\mu(xy)} \mathop{\mathbb{E}}_{\ell} 1_{z_\ell \neq f(x,y)} \mu_\ell(x,y) = \Pr[\pi(x,y) \neq f(x,y)] \leq \epsilon$$

by (40), and $\mathbb{E}_\ell\, b_\ell = 1$, we can bound the second summand in (39) using the Cauchy-Schwarz inequality by

$$\mathop{\mathbb{E}}_{\ell} \sqrt{a_\ell b_\ell} \leq \sqrt{\mathop{\mathbb{E}}_{\ell} a_\ell \mathop{\mathbb{E}}_{\ell} b_\ell} \leq \sqrt{\epsilon}. \tag{42}$$

Using (39), (40), (42), and the monotonicity of $\overline{h}$, we have

$$\begin{aligned} \mathrm{IC}_\mu(f, 0) - \mathrm{IC}_\mu(\pi) &\leq \mathrm{IC}_\mu(\pi') - \mathrm{IC}_\mu(\pi) \leq 2|\mathcal{X} \times \mathcal{Y}| \overline{h}(\epsilon) + 2|\mathcal{X} \times \mathcal{Y}| \overline{h}(\sqrt{\epsilon}) \\ &\leq 4|\mathcal{X} \times \mathcal{Y}| \overline{h}(\sqrt{\epsilon}). \end{aligned}$$

◀

4.1.3 Proof of Proposition 6

Proposition 6 (restated). *Let μ be the distribution defined as*

$$\mu = \begin{array}{|c|c|} \hline 1/2 & 0 \\ \hline 0 & 1/2 \\ \hline \end{array}.$$

Then $\mathrm{IC}^{ext}_\mu(\mathrm{XOR}, \epsilon) \geq \mathrm{IC}^{ext}_\mu(\mathrm{XOR}, 0) - 3\epsilon$.

Proof of Proposition 6. The distribution μ is supported on the inputs $(0,0),(1,1)$, on which the output is 0. It is easy to check (and follows from the analysis below) that $\mathrm{IC}^{ext}_{\mu}(\mathrm{XOR},0) = 1$, since at the end of any protocol that performs $[\mathrm{XOR},0]$, we know whether the input is $(0,0)$ or $(1,1)$.

Consider a protocol π having at most ϵ error on every input, where $\epsilon \le 1/3$. Let $\mathcal{L}_z$ be the set of transcripts on which the output is z; Every transcript is either in $\mathcal{L}_0$ or $\mathcal{L}_1$.

For each transcript t achievable from the initial distribution, the distribution of $XY|t$ is of the form $\begin{array}{|c|c|}\hline p & 0 \\ \hline 0 & 1-p \\ \hline \end{array}$ for some $p = p(t)$. Bayes' law shows that

$$\Pr[t|00] = \frac{\Pr[00|t]\Pr[t]}{\Pr[00]} = 2p(t)\Pr[t], \quad \Pr[t|11] = \frac{\Pr[11|t]\Pr[t]}{\Pr[11]} = 2(1-p(t))\Pr[t].$$

For each transcript t, the rectangle property says $\Pr[t|00]\Pr[t|11] = \Pr[t|10]\Pr[t|01]$. Hence

$$\frac{\Pr[t|01]+\Pr[t|10]}{2} \ge \sqrt{\Pr[t|01]\Pr[t|10]} = \sqrt{\Pr[t|00]\Pr[t|11]} = 2\sqrt{p(t)(1-p(t))}\Pr[t].$$

The protocol π has distributional error at most ϵ, and so

$$\Pr[\mathcal{L}_1] = \sum_{t\in\mathcal{L}_1}\Pr[t] \le \epsilon, \quad \textit{and} \quad \Pr[\mathcal{L}_0] = \sum_{t\in\mathcal{L}_0}\Pr[t] \ge 1-\epsilon.$$

On the other hand, since π has point-wise error at most ϵ, we have

$$\sum_{t\in\mathcal{L}_0}\sqrt{p(t)(1-p(t))}\Pr[t] \le \frac{1}{2}\sum_{t\in\mathcal{L}_0}\frac{\Pr[t|01]+\Pr[t|10]}{2} \le \frac{\epsilon}{2}. \tag{43}$$

Finally,

$$I(XY;\Pi) = H(XY) - H(XY|\Pi) = 1 - \sum_t \Pr[t]h(p(t)).$$

Let T be a random transcript conditioned on belonging to $\mathcal{L}_0$, and consider the random variable $P := p(T)$. On the one hand,

$$1 - I(XY;\Pi) = \sum_t \Pr[t]h(p(t)) \le \Pr[\mathcal{L}_0]\,\mathbb{E}[h(P)] + \Pr[\mathcal{L}_1] \le \mathbb{E}[h(P)] + \epsilon.$$

On the other hand, by (43)

$$\mathbb{E}[\sqrt{P(1-P)}] \le \frac{\epsilon}{2\Pr[\mathcal{L}_0]} \le \frac{\epsilon}{2(1-\epsilon)} \le \epsilon,$$

as we assumed $\epsilon \le 1/3$. Thus it suffices to verify that $\mathbb{E}[h(P)] \le 2\epsilon$ for any random variable P that takes values in $[0,1]$ and satisfies $\mathbb{E}[\sqrt{P(1-P)}] \le \epsilon$. Indeed this would imply

$$1 - I(XY;\Pi) \le \mathbb{E}[h(P)] + \epsilon \le 3\epsilon,$$

alternatively, $\mathrm{IC}^{ext}_{\mu}(\mathrm{XOR},\epsilon) \ge 1-3\epsilon$ for all $\epsilon \le 1/3$, which in turn shows that $\mathrm{IC}^{ext}_{\mu}(\mathrm{XOR},0) = 1$.

Apply the change of variable $Q = \sqrt{P(1-P)}$, so that the assumption simplifies to $\mathbb{E}[Q] \le \epsilon$; note that $0 \le Q \le 1/2$, and $P = (1 \pm \sqrt{1-4Q^2})/2$. Since $h(P) = h(1-P)$, we conclude that

$$\mathbb{E}[h(P)] = \mathbb{E}[\phi(Q)], \text{ where } \phi(Q) = h\left(\frac{1+\sqrt{1-4Q^2}}{2}\right).$$

It is routine to check that the function ϕ is monotonically increasing and strictly convex. Since ϕ is continuous and the domain of Q is restricted to $[0, 1/2]$, the maximum of $\mathbb{E}[\phi(Q)]$ under the constraint $\mathbb{E}[Q] \leq \epsilon$ is achieved[3]. Since ϕ is increasing, the maximum value of $\mathbb{E}[\phi(Q)]$ is achieved when $\mathbb{E}[Q] = \epsilon$. Since ϕ is strictly convex, the maximum value of $\mathbb{E}[\phi(Q)]$ is achieved on a measure supported on the endpoints $0, 1/2$. Thus this measure must be $\Pr[Q = 1/2] = 2\epsilon$ and $\Pr[Q = 0] = 1 - 2\epsilon$. So

$$\mathbb{E}[h(P)] = \mathbb{E}[\phi(Q)] \leq (1 - 2\epsilon)\phi(0) + 2\epsilon\phi(1/2) = 2\epsilon. \qquad \blacktriangleleft$$

4.2 Information complexity with distributional error

Theorem 8 (restated). *Let μ be a probability measure on $\mathcal{X} \times \mathcal{Y}$, and let $f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$ satisfy $\mathrm{IC}_\mu(f, \mu, 0) > 0$. We have*

$$\mathrm{IC}_\mu(f, \mu, 0) - 4|\mathcal{X}||\mathcal{Y}|\overline{h}(\sqrt{\epsilon/\alpha}) \leq \mathrm{IC}_\mu(f, \mu, \epsilon) \leq \mathrm{IC}_\mu(f, \mu, 0) - \frac{\alpha^2}{4} h\left(\epsilon\alpha/4\right) + 3\epsilon \log |\mathcal{X} \times \mathcal{Y}|,$$

where $\alpha = \min_{xy \in \operatorname{supp}\mu} \mu(x, y)$.

Proof of Theorem 8.

Lower bound: The proof is almost identical to the proof of Theorem 7, however now we start from a distribution μ that possibly does not have full support. Consider a protocol π that performs $[f, \mu, \epsilon]$, and define z_ℓ and μ_ℓ as in the proof of Theorem 7. Now the new protocol π' that performs $[f, \mu, 0]$, is defined similar to the one in the proof of Theorem 7 with the only difference that the verification is only performed on the set

$$\Omega'_\ell := \{(x, y) : f(x, y) \neq z_\ell\} \cap \operatorname{supp}\mu.$$

Obviously π' solves $[f, \mu, 0]$. Note that π has point-wise error at most ϵ/α on every point in $\operatorname{supp}\mu$. Thus the same analysis of Theorem 7 shows

$$\mathrm{IC}_\mu(f, \mu, 0) - \mathrm{IC}_\mu(\pi) \leq \mathrm{IC}_\mu(\pi') - \mathrm{IC}_\mu(\pi) \leq 4|\mathcal{X} \times \mathcal{Y}|\overline{h}(\sqrt{\epsilon/\alpha}).$$

Upper bound: For every $z \in \mathcal{Z}$, let $\mathcal{X}_z$ denote the set of all $x \in \mathcal{X}$ such that for some $xy \in \operatorname{supp}\mu$, we have $f(x, y) = z$. Similarly let $\mathcal{Y}_z$ denote the set of all $y \in \mathcal{Y}$ such that for some $xy \in \operatorname{supp}\mu$, we have $f(x, y) = z$. The assumption $\mathrm{IC}_\mu(f, \mu, 0) > 0$ implies the existence of distinct $z_1, z_2 \in \mathcal{Z}$ such that either $\mathcal{X}_{z_1} \cap \mathcal{X}_{z_2} \neq \emptyset$ or $\mathcal{Y}_{z_1} \cap \mathcal{Y}_{z_2} \neq \emptyset$, otherwise, Alice and Bob can exchange the unique values of z determined by their inputs, and since with probability 1, these two values coincide, they can perform $[f, \mu, 0]$ with zero information cost. Hence without loss of generality assume there exists $x_0\overline{y}, x_1\overline{y} \in \operatorname{supp}\mu$ such that $f(x_0, \overline{y}) \neq f(x_1, \overline{y})$ and $\mu(x_0\overline{y}) \geq \mu(x_1\overline{y})$. We will apply Lemma 23. Consider a protocol π with transcript Π that performs $[f, \mu, 0]$, and define the set of transcripts

$$\mathcal{L} := \{t \mid \Pr[x_0\overline{y}|t] \geq \Pr[x_0\overline{y}]/2\},$$

and note that

$$\Pr[x_0\overline{y}] = \sum_t \Pr[x_0\overline{y}|t] \Pr[\Pi = t] \leq \Pr[\Pi \in \mathcal{L}] + \Pr[\Pi \notin \mathcal{L}] \frac{\Pr[x_0\overline{y}]}{2},$$

[3] This follows from Prokhorov's theorem, which implies that the set of probability measures over a $[0, 1/2]$ is compact with respect to the weak-* topology. The same result also follows from the Riesz representation theorem [27].

which implies $\Pr[\Pi \in \mathcal{L}] \geq \frac{\Pr[x_0 \overline{y}]}{2} \geq \frac{\alpha}{2}$. Note that the protocol π' defined in Figure 1 performs $[f, \mu, \epsilon]$. Furthermore we can set $C_1 = C_2 = \alpha/2$ and $\delta = 0$, to obtain

$$\mathrm{IC}_\mu(\pi') \leq \mathrm{IC}_\mu(\pi) - \frac{\alpha^2}{4} h\left(\frac{\epsilon\alpha}{4}\right) + 3\epsilon \log |X \times Y|,$$

for $\epsilon \leq 1/2$. As $-\frac{\alpha^2}{4} h(\epsilon\alpha/4) + 3\epsilon \log |\mathcal{X} \times \mathcal{Y}| \geq 0$ for $\epsilon \geq 1/2$, this finishes the proof for all $0 \leq \epsilon \leq 1$. ◀

4.3 Non-distributional prior-free information cost

In this section we prove Theorem 17, that is

$$\mathrm{IC}(f, \epsilon) \leq \mathrm{IC}(f, 0) - \Omega(h(\epsilon)).$$

First we present some lemmas, and the proof of Theorem 17 will appear at the end of this section.

While Theorem 5 does not give a uniform bound on the parameters C, ϵ_0 for every distribution μ, it does for distributions in which there exist two elements with different outputs, that are in the same row or column and whose probabilities are $\Omega(1)$. We will show that for any non-constant function, the worst distribution is of this form; this might be of independent interest.

We start with the following simple lemma.

▶ **Lemma 24.** *Let $f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$. Suppose that $\operatorname{supp} \mu \subseteq \bigcup_i \mathcal{X}_i \times \mathcal{Y}_i$, where the $\mathcal{X}_i$ and the $\mathcal{Y}_i$ are disjoint. Then*

$$\mathrm{IC}_\mu(f, 0) = \sum_i \mu(\mathcal{X}_i \times \mathcal{Y}_i)\, \mathrm{IC}_{\mu|_{\mathcal{X}_i \times \mathcal{Y}_i}}(f|_{\mathcal{X}_i \times \mathcal{Y}_i}).$$

Proof. The upper bound is easy to see: the players exchange which block they are in, and assuming that they are in the same block, they run an almost optimal protocol for that block. If they are not in the same block, then they exchange inputs, but this happens with probability zero.

In the other direction, let J be the block in which Alice's input lies. Since the value of J is determined by the value of X, for a protocol π with transcript Π, we have

$$I(Y; \Pi | X) = I(Y; \Pi | XJ) = \sum_j \Pr[J = j] I(Y; \Pi | X, J = j) = \sum_j \mu(\mathcal{X}_j \times \mathcal{Y}_j) I(Y; \Pi | X, J = j).$$

With probability 1, J is also the block in which Bob's input lies, and so

$$\begin{aligned}\mathrm{IC}_\mu(\pi) &= \sum_j \mu(\mathcal{X}_j \times \mathcal{Y}_j)[I(X; \Pi | Y, J = j) + I(Y; \Pi | X, J = j)] \\ &\geq \sum_j \mu(\mathcal{X}_j \times \mathcal{Y}_j)\, \mathrm{IC}_{\mu|_{\mathcal{X}_j \times \mathcal{Y}_j}}(f|_{\mathcal{X}_j \times \mathcal{Y}_j}).\end{aligned}$$ ◀

We can therefore restrict our attention (for now) to distributions based on a single block. The crucial observation is the following.

▶ **Lemma 25.** *Let $f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$, and let μ be a distribution such that f is constant on its support, each atom in the support has probability at least α, and the marginals of the support are $\mathcal{X}, \mathcal{Y}$. If f is not constant then there is a distribution ν such that $\mathrm{IC}_\nu(f, 0) \geq \mathrm{IC}_\mu(f, 0) + C(\alpha)$, where $C(\alpha) > 0$ depends only on $\alpha, |\mathcal{X}|, |\mathcal{Y}|$.*

Proof. Let (x_0, y_0) be any point not in the support of μ such that $f(x_0y_0)$ is different from the constant value of f on $\operatorname{supp}\mu$. Since the marginals of the support are $\mathcal{X}, \mathcal{Y}$ and every atom in the support has probability at least α, we see that $\Pr[X = x_0], \Pr[Y = y_0] \geq \alpha$.

Let $\nu = \epsilon\delta_{x_0y_0} + (1-\epsilon)\mu$, where ϵ is a parameter to be determined later, and $\delta_{x_0y_0}$ denotes the Dirac measure concentrated on the point (x_0, y_0). Note that $X'Y' \sim \nu$ can be sampled in the following manner. First we pick $XY \sim \mu$ and an independent Bernoulli random variable B with $\Pr[B = 1] = \epsilon$. Then

$$X'Y' = \begin{cases} XY & \text{if } B = 0, \\ x_0y_0 & \text{if } B = 1. \end{cases}$$

Let π be a protocol that performs the task $[f, 0]$, and let Π_{xy} denote the transcript of this protocol when it is run on the input xy. Note that with probability 1, the value of B is determined by the value of $X'Y'$, and thus

$$\begin{aligned} I(X'; \Pi_{X'Y'}|Y') &= I(X'B; \Pi_{X'Y'}|Y') = I(B; \Pi_{X'Y'}|Y') + I(X'; \Pi_{X'Y'}|Y'B) \\ &= I(B; \Pi_{X'Y'}|Y') + (1-\epsilon)I(X; \Pi_{XY}|Y). \end{aligned}$$

Moreover, since $f(x_0, y_0)$ is different from the constant value of f on the support of μ, the value of B is determined by $\Pi_{X'Y'}$. Thus $I(B; \Pi_{X'Y'}|Y') = H(B|Y')$, and

$$I(X'; \Pi_{X'Y'}|Y') = H(B|Y') + (1-\epsilon)I(X; \Pi_{XY}|Y).$$

To lower-bound $H(B|Y')$, note that

$$\Pr[B = 1|Y' = y_0] = \frac{\Pr[B = 1, Y' = y_0]}{\Pr[Y' = y_0]} = \frac{\epsilon}{(1-\epsilon)\Pr[Y = y_0] + \epsilon} \geq \epsilon,$$

and on the other hand,

$$\Pr[B = 1|Y' = y_0] \leq \frac{\epsilon}{(1-\epsilon)\alpha + \epsilon},$$

which for $\epsilon \leq \sqrt{\alpha}/2$ will be at most $1 - \epsilon$. Since $\Pr[Y' = y_0] = (1-\epsilon)\Pr[Y = y_0] + \epsilon \geq \alpha$, we conclude that $H(B|Y') \geq \alpha h(\epsilon)$. We deduce that

$$I(X'; \Pi_{X'Y'}|Y') \geq \alpha h(\epsilon) + (1-\epsilon)I(X; \Pi_{XY}|Y) \geq I(X; \Pi_{XY}|Y) + \alpha h(\epsilon) - \epsilon \log|\mathcal{X} \times \mathcal{Y}|.$$

The gain is

$$I(X'; \Pi_{X'Y'}|Y') - I(X; \Pi_{XY}|Y) \geq \alpha\epsilon \log\frac{1}{\epsilon} - \epsilon \log|\mathcal{X} \times \mathcal{Y}| = \left(\alpha \log\frac{1}{\epsilon} - \log|\mathcal{X} \times \mathcal{Y}|\right)\epsilon,$$

and so when $\epsilon \leq \epsilon_0 := |\mathcal{X} \times \mathcal{Y}|^{-2/\alpha}$, the gain is at least $\epsilon \log|\mathcal{X} \times \mathcal{Y}|$. Taking $\epsilon = \min(\epsilon_0, \sqrt{\alpha}/2)$, we obtain a constant $C(\alpha) > 0$, depending on $|\mathcal{X} \times \mathcal{Y}|$, such that

$$I(X'; \Pi_{X'Y'}|Y') \geq I(X; \Pi_{XY}|Y) + C(\alpha),$$

and similarly $I(Y'; \Pi_{X'Y'}|X') \geq I(X; \Pi_{XY}|Y) + C(\alpha)$. This shows that

$$\mathrm{IC}_\nu(f, 0) \geq \mathrm{IC}_\mu(f, 0) + 2C(\alpha). \qquad \blacktriangleleft$$

We obtain the following important consequence.

▶ **Lemma 26.** *Let $f\colon \mathcal{X}\times\mathcal{Y}\to\mathcal{Z}$ be a non-constant function. There exist constants $c,\delta>0$, depending only on the function f and $|\mathcal{X}|,|\mathcal{Y}|$, such that if $\mathrm{IC}_\mu(f,0)\geq \mathrm{IC}(f,0)-\delta$ then there exist points P,Q, on the same row or column, such that $\mu(P),\mu(Q)\geq c$ and $f(P)\neq f(Q)$.*

Proof. Call a distribution ν on $\mathcal{X}\times\mathcal{Y}$ *optimal* if $\mathrm{IC}(f,0)=\mathrm{IC}_\nu(f,0)$. Braverman et al. [5] showed that $\mathrm{IC}_\nu(f,0)$ is continuous in ν, and this implies that optimal distributions exist, and moreover the set of optimal distributions is closed. It is also convex, due to the concavity of $\mathrm{IC}_\nu(f,0)$ (see [4]).

For a distribution ν, let $\beta(\nu)$ be the maximal value β such that there exist two points P,Q, on the same row or column, such that $\nu(P),\nu(Q)\geq\beta$ and $f(P)\neq f(Q)$. Note that $\beta(\nu)$ is continuous in ν.

Suppose that $\beta(\nu)=0$. For $z\in\mathcal{Z}$, let $\mathcal{X}_z$ be the set of rows on which some point $P\in\operatorname{supp}\nu$ satisfies $f(P)=z$, and define $\mathcal{Y}_z$ analogously. We claim that the sets $\mathcal{X}_z$ for $z\in\mathcal{Z}$ are disjoint, similarly $\mathcal{Y}_z$ are disjoint. Indeed, if $x\in\mathcal{X}_{z_1}\cap\mathcal{X}_{z_2}$, then the row x contains two points P,Q in the support such that $f(P)\neq f(Q)$, and so $\beta(\nu)>0$. Next we show that $\operatorname{supp}\nu\subseteq\bigcup_z\mathcal{X}_z\times\mathcal{Y}_z$. Indeed if $P\in\mathcal{X}_{z_1}\times\mathcal{Y}_{z_2}$ is in the support of ν, and $f(P)\neq z_1$, then there exists some point Q on the same row as P is in the support and satisfies $f(Q)=z_1$, showing that $\beta(\nu)>0$; a similar conclusion is reached if $f(P)\neq z_2$.

Consider now one of the blocks $\mathcal{X}_z\times\mathcal{Y}_z$. Lemma 25 shows that we can modify the component of ν on that block so as to increase the information complexity, and Lemma 24 shows that this increases the information complexity over the entire domain. We conclude that ν is not optimal.

For $\rho\geq 0$, let $O_\rho=\{\nu:\mathrm{IC}_\nu(f,0)\geq\mathrm{IC}(f,0)-\rho\}$. Continuity of $\mathrm{IC}_\nu(f,0)$ shows that O_ρ is closed. We define $b(\rho)=\inf\{\beta(\nu):\nu\in O_\rho\}$; since β is continuous and O_ρ is closed, the infimum is achieved. In view of the preceding paragraph, $b(0)>0$. Continuity of $\beta(\nu)$ and $\mathrm{IC}_\nu(f,0)$ shows that $b(\rho)$ is continuous as well, and so $b(\delta)>0$ for some $\delta>0$. The proof is complete by taking $c=b(\delta)$. ◀

We can now apply Theorem 5 to deduce that $\mathrm{IC}(f,\epsilon)\leq\mathrm{IC}(f,0)-\Omega(h(\epsilon))$.

Theorem 17 (restated). *If $f\colon\mathcal{X}\times\mathcal{Y}\to\mathcal{Z}$ is non-constant then*

$$\mathrm{IC}(f,\epsilon)\leq\mathrm{IC}(f,0)-\Omega(h(\epsilon)),$$

where the hidden constant depends on f.

Proof. Let c,δ be the parameters from Lemma 26. For a distribution μ, either $\mathrm{IC}_\mu(f,0)\leq\mathrm{IC}(f,0)-\delta$ or Theorem 5 shows that $\mathrm{IC}_\mu(f,\epsilon)\leq\mathrm{IC}_\mu(f,0)-(c^3/32)h(\epsilon)\leq\mathrm{IC}(f,0)-(c^3/32)h(\epsilon)$ for all $\epsilon\leq\epsilon_0$ where ϵ_0 depends only on c and $|\mathcal{X}\times\mathcal{Y}|$. Choose ϵ sufficiently enough such that $(c^3/32)h(\epsilon)\leq\delta$ and $\epsilon\leq\epsilon_0$, we conclude in both cases that $\mathrm{IC}_\mu(f,\epsilon)\leq\mathrm{IC}(f,0)-\Omega(h(\epsilon))$. ◀

4.4 A characterization of trivial measures

First we present the proof of the external case, i.e. Theorem 22, as it is simpler.

Theorem 22 (restated). *Let $f\colon\mathcal{X}\times\mathcal{Y}\to\mathcal{Z}$ be an arbitrary function, and μ a distribution on $\mathcal{X}\times\mathcal{Y}$. The distribution μ is external-trivial iff it is strongly external-trivial iff it is structurally external-trivial.*

Proof of Theorem 22.

If μ is external-trivial then μ is structurally external-trivial. Suppose that μ is external-trivial but not structurally external-trivial. We will reach a contradiction.

We start by showing that if μ is external-trivial then f has to be constant on the support of μ. Indeed, suppose that the protocol π computes f correctly, and denote by Π the transcript of π. The data processing inequality shows that

$$I(\Pi; XY) \geq I(\Pi; f(XY)) = H(f(XY)) - H(f(XY)|\Pi) = H(f(XY)).$$

This shows that μ can only be external-trivial if $H(f(XY)) = 0$, that is, if f is constant on the support of μ. From now, we assume that this is indeed the case.

Let ab be an arbitrary point in the support of μ, and let $c = f(ab)$. Since μ is not structurally external-trivial, there must be some input $x_0y_0 \in S_A \times S_B$ for which $f(x_0y_0) \neq c$. Note that x_0y_0 is not in the support of μ. Since $x_0 \in S_A$, x_0y_1 is in the support of μ for some $y_1 \in S_B$. Similarly, x_1y_0 is in the support of μ for some $x_1 \in S_A$.

Since μ is external-trivial, there is a sequence π_n of protocols computing f correctly on every input such that $I(XY; \Pi_n) \to 0$, where $XY \sim \mu$. We think of π_n also as a distribution over transcripts t. Since $f(XY) = c$ with probability 1, if $\pi_n(t) > 0$ then the transcript t indicates that the output is c. Let p_n be the joint distribution of X, Y, t. Recall that $D(p_n(x,y,t)\|\mu(x,y)\pi_n(t)) = I(XY; \Pi_n)$, hence $D(p_n(x,y,t)\|\mu(x,y)\pi_n(t)) \to 0$.

For two distributions μ and ν on a finite space, Pinsker's inequality states that $D(\mu||\nu) \geq \frac{1}{2}\|\mu - \nu\|_1^2$. This implies that $\|p_n(x,y,t) - \mu(x,y)\pi_n(t)\|_1 \to 0$. On the other hand, for every transcript t appearing with positive probability, either $p_n(x_0, y_1, t) = 0$ or $p_n(x_1, y_0, t) = 0$: otherwise $p_n(x_0, y_0, t) > 0$ (due to the rectangular property of protocols), contradicting the correctness of π_n (since $f(x_0y_0) \neq c$). Therefore

$$|\mu(x_0,y_1)\pi_n(t) - p_n(x_0,y_1,t)| + |\mu(x_1,y_0)\pi_n(t) - p_n(x_1,y_0,t)| \geq \pi_n(t)\min(\mu(x_0,y_1), \mu(x_1,y_0)).$$

Summing over all transcripts having positive probability, we deduce that

$$\|p_n(x,y,t) - \mu(x,y)\pi_n(t)\|_1 \geq \sum_t \pi_n(t)\min(\mu(x_0,y_1), \mu(x_1,y_0)) = \min(\mu(x_0,y_1), \mu(x_1,y_0)),$$

contradicting our assumption that $\|p_n(x,y,t) - \mu(x,y)\pi_n(t)\|_1 \to 0$.

If μ is structurally external-trivial then μ is strongly external-trivial. Consider the following protocol. Alice tells Bob whether her input is in S_A. Bob tells Alice whether his input is in S_B. If the input is in $S_A \times S_B$, then the output is known. Otherwise, the players reveal their inputs (but this happens with probability zero). It's not difficult to check that this protocol has zero external information cost.

If μ is strongly external-trivial then μ is external-trivial. This is obvious. ◀

We comment that our proof gives an explicit lower bound on $\mathrm{IC}_\mu^{ext}(f, 0)$ whenever μ is not external-trivial.

Next we present the proof of Theorem 19, showing that all our definitions of internal triviality are equivalent. As before, we can get an explicit lower bound on $\mathrm{IC}_\mu(f, 0)$ whenever μ is not internal-trivial.

Theorem 19 (restated). *Let $f\colon \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$ be an arbitrary function, and μ a distribution on $\mathcal{X} \times \mathcal{Y}$. The distribution μ is internal-trivial iff it is strongly internal-trivial iff it is structurally internal-trivial.*

Proof of Theorem 19.

If μ is internal-trivial then μ is structurally internal-trivial. Suppose that μ is internal-trivial but not structurally internal-trivial. We will reach a contradiction.

Since μ is internal-trivial, there is a sequence of protocols π_n such that $I(X;\Pi_n|Y) + I(Y;\Pi_n|X) \to 0$. In particular, $I(X;\Pi_n|Y), I(Y;\Pi_n|X) \to 0$. Moreover, for every $x \in S_A$ and for every $y \in S_B$, $I(X;\Pi_n|Y=y), I(Y;\Pi_n|X=x) \to 0$.

Let $p_n(x,y,t)$ be the joint probability of the input and of the transcript of π_n being t. We also think of π_n as a distribution over transcripts. As in the proof of Theorem 22, using Pinsker's inequality we deduce that for all $y \in S_B$, $\|p_n(x,t|y) - \mu(x|y)\pi_n(t|y)\|_1 \to 0$, and so for all $y \in S_B$,

$$B_y := \sum_{x,t} |p_n(x,y,t) - \mu(x,y)\pi_n(t|y)| \to 0.$$

Similarly, for all $x \in S_A$ we have

$$A_x := \sum_{y,t} |p_n(x,y,t) - \mu(x,y)\pi_n(t|x)| \to 0.$$

According to Lemma 21, there exists a connected component C of G_μ such that f is not constant on $C_A \times C_B$. Suppose first that there is an edge (P,Q) on which f is not constant. Without loss of generality, assume $P = (a,y_0)$ and $Q = (a,y_1)$. Thus

$$\sum_t |p_n(a,y_0,t) - \mu(a,y_0)\pi_n(t|a)| + |p_n(a,y_1,t) - \mu(a,y_1)\pi_n(t|a)| \to 0.$$

On the other hand, for each transcript t either $p_n(a,y_0,t) = 0$ or $p_n(a,y_1,t) = 0$, since $f(ay_0) \neq f(ay_1)$. Thus

$$\sum_t |p_n(a,y_0,t) - \mu(a,y_0)\pi_n(t|a)| + |p_n(a,y_1,t) - \mu(a,y_1)\pi_n(t|a)| \geq \sum_t \pi_n(t|a)\min(\mu(a,y_0),\mu(a,y_1)) = \min(\mu(a,y_0),\mu(a,y_1)),$$

contradicting the assumption that the left-hand side tends to zero.

Suppose next that f is constant across all edges (and so on the entire connected component), say $f(x,y) = c$ for all $(x,y) \in C$. Since f is not monochromatic on $C_A \times C_B$, there must exist a point $P \in C_A \times C_B$ such that $f(P) \neq c$. There must be points $P_A, P_B \in \operatorname{supp}\mu$ with the same row and column (respectively) as P. Since P_A, P_B are in the same connected component, there is some path $P_A = Q_0, Q_1, \ldots, Q_m = P_B$ connecting them: for every $i < m$, Q_i, Q_{i+1} are either in the same row or in the same column. We can assume that $m \leq M := |\mathcal{X}| + |\mathcal{Y}|$. No transcript can have positive probability for both Q_0 and Q_m, since otherwise it would have positive probability for P as well, and this cannot happen since $f(Q_0) = f(Q_m) = c$ while $f(P) \neq c$.

Let t be any transcript satisfying $p_n(Q_0,t) > 0$. Since $p_n(Q_m,t) = 0$, there must be an index i such that $p_n(t|Q_i) - p_n(t|Q_{i+1}) \geq p_n(t|Q_0)/m \geq p_n(t|Q_0)/M$. Assume without loss of generality that $Q_i = (a,y_0)$ and $Q_{i+1} = (a,y_1)$. The contribution of t to A_a is

$$\begin{aligned}
&|\mu(a,y_0)\pi_n(t|a) - p_n(a,y_0,t)| + |\mu(a,y_1)\pi_n(t|a) - p_n(a,y_1,t)| = \\
&\mu(a,y_0)|\pi_n(t|a) - p_n(t|a,y_0)| + \mu(a,y_1)|\pi_n(t|a) - p_n(t|a,y_1)| \geq \\
&\qquad \frac{\min(\mu(a,y_0),\mu(a,y_1))}{M} p_n(t|Q_0) \geq \frac{\min(\mu(a,y_0),\mu(a,y_1))}{M} p_n(Q_0,t),
\end{aligned}$$

using the triangle inequality in the form $|\alpha-\gamma| + |\gamma-\beta| \geq |\alpha-\beta|$.

Denoting by δ the minimum of $\mu(x,y)$ over the support of μ, we conclude that $\sum_x A_x + \sum_y B_y$ is at least

$$\sum_t \frac{\delta}{M} p_n(Q_0,t) = \frac{\delta}{M}\mu(Q_0) \geq \frac{\delta^2}{M},$$

contradicting our assumption that $\sum_x A_x + \sum_y B_y \to 0$.

If μ is structurally internal-trivial then μ is strongly internal-trivial. Consider the following protocol. Alice tells Bob which block $\mathcal{X}_i$ her input belongs to. Bob tells Alice which block $\mathcal{Y}_i$ his input belongs to. If the input is in $\mathcal{X}_i \times \mathcal{Y}_i$, then the output is known. Otherwise, the players reveal their inputs (but this happens with probability zero). It's not difficult to check that this protocol has zero internal information cost.

If μ is strongly internal-trivial then μ is internal-trivial. This is obvious. ◀

5 Parametrization of all distributions as product distributions

In Section 2.5 we discussed how a communication protocol can be interpreted as a random walk on the set of distributions on $\mathcal{X} \times \mathcal{Y}$. Every time a player sends a signal, we update the underlying distribution based on the information provided by the sent signal. These updates are by scaling either the $\mathcal{X}$ marginal or the $\mathcal{Y}$ marginal of the distribution. This restricted way in which the underling distribution can be updated will allow us to parametrize the set of all reachable distributions from a specific distribution $\overline{\mu}$ in such a way that the changes are captured by product measures. First note that each reachable distribution $\overline{\mu}'$ can be identified by the constants that multiplied $\overline{\mu}$ to obtain $\overline{\mu}'$.

To formalize this intuition, we have the following definition.

▶ **Definition 27.** For two distributions $\mu, \nu \in \Delta(\mathcal{X},\mathcal{Y})$, define

$$\mu \odot \nu := \frac{\mu \cdot \nu}{\langle \mu, \nu \rangle}, \tag{44}$$

where $\mu \cdot \nu$ is the usual point-wise product of the two measures.

Clearly, $\mu \odot \nu \in \Delta(\mathcal{X},\mathcal{Y})$ unless $\langle \mu, \nu \rangle = 0$, in which case the product is undefined. For our purposes, we will consider decompositions of the form $\overline{\mu} = \nu \odot \mu$, where μ is a *product measure*. The statement "$\overline{\mu}$ is a distribution obtained from ν by scaling its rows and columns" is equivalent to "there exists a product measure μ such that $\overline{\mu} = \nu \odot \mu$". Note that if μ is the uniform distribution, then $\nu = \mu \odot \nu$ for all distributions ν.

Let $\overline{\mu}$ be the prior distribution on $\mathcal{X} \times \mathcal{Y}$ in a communication protocol. We fix a decomposition $\overline{\mu} = \nu \odot \mu$, where μ is a product distribution. For every distribution $\overline{\mu}'$ reachable from $\overline{\mu}$ there is a product distribution μ' such that $\overline{\mu}' = \nu \odot \mu'$, for the same distribution ν. This follows from the fact that $\overline{\mu}'$ is obtained from $\overline{\mu}$ by scaling its rows and columns; therefore if we scale the rows and columns of μ by the same constants and then normalize it, we obtain the desired μ'. In such a decomposition $\overline{\mu} = \nu \odot \mu$, $\overline{\mu}$ is called the *real* distribution, ν the *reference distribution* and μ the *pretend distribution*.

We would like to work with product distributions since they are simpler, and easier to analyze, as we will demonstrate in Section 6. Therefore, we define a *pretend random walk*, which is a random walk on pretend distributions, as opposed to the normal random walk presented in Section 2.5, which we call the *real random walk* to distinguish it from

the pretend one. It starts from a product measure $\mu = (\mu^{\mathcal{X}}, \mu^{\mathcal{Y}})$, where $\mu^{\mathcal{X}}$ and $\mu^{\mathcal{Y}}$ are the $\mathcal{X}$ and $\mathcal{Y}$ marginals of μ. At each step we either move by scaling the $\Delta(\mathcal{X})$ marginal or the $\Delta(\mathcal{Y})$ marginal. The transition in $\Delta(\mathcal{X})$ is performed by moving with probability λ_0 to $(\mu_0, \mu^{\mathcal{Y}})$ and with probability λ_1 to $(\mu_1, \mu^{\mathcal{Y}})$, where $0 < \lambda_0, \lambda_1 < 1$, $\lambda_0 + \lambda_1 = 1$ and $\sum_{b=0,1} \lambda_i \mu_i = \mu^{\mathcal{X}}$. A step in the $\Delta(\mathcal{Y})$ direction is performed similarly.

Every pretend random walk corresponds to a real random walk performed by some protocol. Given such a pretend random walk, and a reference distribution ν, if we replace every distribution μ encountered in the random walk by $\nu \odot \mu$, and scale the transition probabilities, we obtain a real random walk performed by some protocol. Here ν can be any distribution such that $\nu \odot \mu$ is defined for every μ encountered in the protocol (e.g. if $\operatorname{supp} \nu$ includes the support of the initial distribution). The inverse transformation is also possible.

To formalize this idea, consider a pretend random walk step, from μ to μ_0 and μ_1 with transition probabilities λ_0 and λ_1, respectively. Fix a reference distribution ν. Then

$$\nu \odot \mu = \frac{\nu \cdot \mu}{\langle \nu, \mu \rangle} = \sum_{b=0,1} \lambda_b \frac{\nu \cdot \mu_b}{\langle \nu, \mu \rangle} = \sum_{b=0,1} \frac{\langle \nu, \mu_b \rangle}{\langle \nu, \mu \rangle} \lambda_b (\nu \odot \mu_b) = \sum_{b=0,1} \overline{\lambda_b} (\nu \odot \mu_b)$$

for the values

$$\overline{\lambda_b} = \frac{\langle \nu, \mu_b \rangle}{\langle \nu, \mu \rangle} \lambda_b. \tag{45}$$

A calculation shows

$$\sum_{b=0,1} \overline{\lambda_b} = \sum_{b=0,1} \frac{\langle \nu, \mu_b \rangle}{\langle \nu, \mu \rangle} \lambda_b = \frac{\langle \nu, \sum_{b=0,1} \lambda_b \mu_b \rangle}{\langle \nu, \mu \rangle} = \frac{\langle \nu, \mu \rangle}{\langle \nu, \mu \rangle} = 1.$$

Furthermore, if the pretend random walk step is performed in the $\Delta(\mathcal{X})$ direction, then $\nu \odot \mu_b$ is obtained by scaling the rows of μ, and if in the $\Delta(\mathcal{Y})$ direction, then by scaling the columns. Therefore, there exists a real random walk step where we move from $\nu \odot \mu$ to $\nu \odot \mu_0$ and $\nu \odot \mu_1$ with probabilities $\overline{\lambda_0}$ and $\overline{\lambda_1}$ respectively. The conversion in the opposite direction, from the real world to the pretend world, is possible due to essentially the same calculations.

Let π_0 and π_1 be the two branches of the protocol π corresponding to the value of the first bit that was sent. Let $\overline{\mu}$ be an input distribution that moves either to $\overline{\mu}_0$ or to $\overline{\mu}_1$ with probabilities $\overline{\lambda_0}$ and $\overline{\lambda_1}$, respectively. The following equation regarding the concealed information,

$$\mathrm{CI}_{\overline{\mu}}(\pi) = \sum_{b=0,1} \overline{\lambda_b} \, \mathrm{CI}_{\overline{\mu}_b}(\pi_b)$$

translates to

$$\mathrm{CI}_{\nu \odot \mu}(\pi) = \sum_{b=0,1} \frac{\langle \nu, \mu_b \rangle}{\langle \nu, \mu \rangle} \lambda_b \, \mathrm{CI}_{\nu \odot \mu_b}(\pi_b).$$

Multiplying by $\langle \nu, \mu \rangle$ we get

$$\mathrm{CI}_{\nu \odot \mu}(\pi) \langle \nu, \mu \rangle = \sum_{b=0,1} \lambda_b \langle \nu, \mu_b \rangle \, \mathrm{CI}_{\nu \odot \mu_b}(\pi_b),$$

This motivates the following definition.

▶ **Definition 28.** Let ν be a fixed reference distribution. Define the *scaled information* of a protocol π with respect to a product distribution μ as

$$\mathrm{SIM}_\mu(\pi) := \langle \nu, \mu \rangle \, \mathrm{CI}_{\nu \odot \mu}(\pi). \tag{46}$$

Equation (46) allows us to write

$$\mathrm{SIM}_\mu(\pi) = \lambda_0 \, \mathrm{SIM}_{\mu_0}(\pi_0) + \lambda_1 \, \mathrm{SIM}_{\mu_1}(\pi_1). \tag{47}$$

Recall that CI is the expected amount of entropy that the players have concealed from each other by the end of the protocol. To formally state this, let $\overline{\mu}$ be a distribution over the inputs, π some protocol and Π the random variable representing the transcript of the protocol. Let $\overline{\mu}_\Pi$ be the random variable that represents the distribution over the inputs given the transcript Π, as defined in Section 2.5. Then

$$\mathrm{CI}_{\overline{\mu}}(\pi) = \mathop{\mathbb{E}}_{\Pi} \left[H_{\overline{\mu}_\Pi}(X|Y) + H_{\overline{\mu}_\Pi}(Y|X) \right]. \tag{48}$$

We will translate (48) to a formula involving the pretend random walk. Let $\overline{\mu} = \nu \odot \mu$, and denote by μ_Π the pretend distribution where the pretend random walk ends if its associated protocol has the transcript Π. Or, in a more formal way, μ_Π is the distribution such that $\nu \odot \mu_\Pi = \overline{\mu}_\Pi$. Equation (46) implies

$$\mathrm{SIM}_\mu(\pi) = \mathop{\mathbb{E}}_{\Pi} \langle \nu, \mu_\Pi \rangle \left[H_{(\nu \odot \mu)_\Pi}(X|Y) + H_{(\nu \odot \mu)_\Pi}(Y|X) \right], \tag{49}$$

where the probability for each transcript Π is according to the pretend random walk rather than the real one.

One should ask: What is the probability of a transcript t in the pretend random walk, given its probability $\overline{\lambda}$ in the real world? The answer turns out to be very simple. Let $\overline{\mu}^0, \ldots, \overline{\mu}^k$ be the real distributions encountered in the real random walk, where $\overline{\mu}^0$ is the input distribution and $\overline{\mu}^k = \overline{\mu}_t$ is the last distribution encountered. For all $1 \le i \le k$, let $\overline{\lambda^i}$ be the transition probability from $\overline{\mu}^{i-1}$ to $\overline{\mu}^i$ in the real random walk, so that $\overline{\lambda} = \overline{\lambda^1} \cdots \overline{\lambda^k}$. Let μ^i be the pretend distribution associated with $\overline{\mu}^i$ such that $\overline{\mu}^i = \nu \odot \mu^i$ for all i . Then, the transition probability from μ^{i-1} to μ^i in the pretend world equals

$$\lambda^i = \frac{\langle \nu, \mu^{i-1} \rangle}{\langle \nu, \mu^i \rangle} \overline{\lambda^i},$$

using the conversion in (45). Multiplying all together, we get that the probability of t in the pretend world is

$$\lambda = \prod_{i=1}^{k} \lambda^i = \prod_{i=1}^{k} \frac{\langle \nu, \mu^{i-1} \rangle}{\langle \nu, \mu^i \rangle} \overline{\lambda^i} = \frac{\langle \nu, \mu^0 \rangle}{\langle \nu, \mu^k \rangle} \overline{\lambda}.$$

This equation also shows how one can derive (49) from (48) by multiplying the equation by $\langle \nu, \mu^0 \rangle$.

6 The analysis of the AND function

This section is mainly devoted to proving the only remaining case of Theorem 9, i.e. the lower bound on $\mathrm{IC}_\mu(\mathrm{AND}, \epsilon)$. This is presented below separately as Theorem 33. Our general strategy for this proof was sketched in Section 3.3 following Theorem 9.

Table 1 The leaf distribution of the buzzer protocol starting from (p, q), where $p \geq q$.

Distribution μ_Π	$(p, 0)$	$(\ell, 0)$, $(0, \ell)$ $(p < \ell < 1)$	$(1, 1)$
The probability to reach that distribution	$1 - q/p$	$pq/\ell^3 \, \mathrm{d}\ell$	pq

Preliminaries and notations. The section relies strongly on the parametrization of distributions as product distributions, as presented in Section 5. A real distribution is usually denoted as $\overline{\mu}$, and it is usually decomposed as $\overline{\mu} = \nu \odot \mu$, where ν is a symmetric reference distribution and μ a pretend distribution. Pretend distributions are always product ones. We will use the shorthand notation $\mu = (p, q)$ for the product distribution in which $p = \mu(1, 0) + \mu(1, 1)$ and $q = \mu(0, 1) + \mu(1, 1)$. The distribution $\overline{\mu}$ will usually be assumed to be of full support, which in turn forces ν and μ to be so too.

We are usually going to be working in a pretend world, dealing with the pretend distributions, and keeping the reference distributions in the background. Furthermore, reference distributions are usually kept fixed. We regard protocols as pretend random walks, as presented in Section 5.

Suppose that we run a protocol π starting at a distribution $\overline{\mu} = \nu \odot \mu$. As we explained in Section 5, for each transcript t of the protocol, there is a product distribution μ_t such that $\nu \odot \mu_t$ is the distribution of the players' inputs conditioned on the protocol terminating at the leaf t. Let Π be the random transcript of the pretend random walk associated with an execution of π on input distribution $\overline{\mu}$. Therefore, for any transcript t, $\Pr[\Pi = t]$ is the probability for the transcript t in the pretend random walk, which might be different than the corresponding probability in the real random walk. Throughout this section our view of the protocol is only by the pretend random walk, therefore all random variable that correspond to Π are assumed to be distributed according to the pretend random walk. Since μ_Π, the pretend distribution on the random transcript Π, is a product distribution, it can be written as $\mu_\Pi = (\mathbf{p}, \mathbf{q})$, where $\mathbf{p}, \mathbf{q}$ are random variables. We call $(\mathbf{p}, \mathbf{q})$ the *leaf distribution* of π. We define a crucial random variable, $\boldsymbol{\ell} = \max(\mathbf{p}, \mathbf{q})$.

If π is a zero-error protocol, then the leaf distribution is supported on product distributions of the form $(p, 0)$, $(0, q)$ or $(1, 1)$, since in order to know the AND of the two players' inputs we need to know that one of the players has input 0, or that both inputs are 1.

Since we are concerned with almost-optimal protocol, we would like to quantify optimality. Given a protocol π, define its *wastage* with respect to a distribution $\overline{\mu}$ by

$$\mathrm{IW}_{\overline{\mu}}(\pi) = \mathrm{IC}_{\overline{\mu}}(\pi) - \mathrm{IC}_{\overline{\mu}}(\mathrm{AND}, 0) = \mathrm{CI}_{\overline{\mu}}(\mathrm{AND}, 0) - \mathrm{CI}_{\overline{\mu}}(\pi).$$

6.1 Stability results

Braverman et al. [4], studying the complexity of the AND function, suggested a continuous protocol whose information complexity equals $\mathrm{IC}_{\overline{\mu}}(\mathrm{AND}, 0)$, called the *buzzer protocol*. This protocol is defined differently for any input distribution $\overline{\mu}$. Here we denote this protocol by π^*. The buzzer protocol is not a conventional communication protocol as it has access to a continuous clock, however, it can be viewed as a limit of a sequence of genuine protocols. The information complexity of the protocols in that sequence converges to that of the buzzer protocol, and their leaf distribution converges in distribution.

We start by presenting the leaf distribution of the buzzer protocol. We assume that the input reference distribution is symmetric; its importance will become apparent later on.

As it can be seen in Table 1, this is a mix of discrete probabilities and a continuous density. To verify that the above formulas are correct, we can convert the leaf distribution of

the buzzer protocol as it is calculated in [4] for the real random walk to its corresponding leaf distribution in the pretend random walk. The formulas that are discussed in Section 5 can be used to calculate the appropriate scaling of the probabilities as we convert the real random walk to the pretend one.

There is also a second and more intuitive way to obtain these formulas. This is done by considering a sequence of protocols that converges to the buzzer protocol. We describe the protocols in that sequence by their pretend random walk. The initial distribution in the pretend world of a protocol in that sequence is (p, q), where $p, q \in \{0, \frac{1}{n}, \frac{2}{n}, \ldots, 1\}$. In each step, the pretend random walk moves to one of two adjacent grid points, each with probability half. If we are currently in a distribution $(\frac{a}{n}, \frac{b}{n})$ where $a \geq b$, then the step moves to one of $(\frac{a}{n}, \frac{b+1}{n})$ and $(\frac{a}{n}, \frac{b-1}{n})$. Otherwise, the protocol moves to one of $(\frac{a+1}{n}, \frac{b}{n})$ and $(\frac{a-1}{n}, \frac{b}{n})$.

Therefore, starting at the point $(\frac{a}{n}, \frac{b}{n})$ where $a \geq b$, the random walk moves in the y axis, until it ends up either at $(\frac{a}{n}, 0)$ or at $(\frac{a}{n}, \frac{a+1}{n})$. Since this walk is balanced, the probabilities to get to these points are $1 - \frac{b}{a+1}$ and $\frac{b}{a+1}$, respectively. Then, from that point the random walk moves in the x axis, until it either gets to the point $(0, \frac{a+1}{n})$ or to $(\frac{a+1}{n}, \frac{a+1}{n})$, with probabilities $\frac{1}{a+1}$ and $\frac{a}{a+1}$ respectively. Then again, it ends up either at $(\frac{a+1}{n}, 0)$ or at $(\frac{a+1}{n}, \frac{a+2}{n})$, then at $(0, \frac{a+2}{n})$ or $(\frac{a+2}{n}, \frac{a+2}{n})$ and continues this way, until it either gets to the point $(1, 1)$, or to a point of the form $(0, \frac{i}{n})$ or $(\frac{i}{n}, 0)$. Calculating the leaf distribution of each pretend random walk in that sequence, and taking the limit as $n \to \infty$, results in a leaf distribution, which equals that of the buzzer protocol, as will be explained below.

The buzzer protocol can also be defined similarly as a sequence of converging protocols, where for each protocol in the sequence, the real-world analogue of moving in the y direction is performed whenever $\Pr[X = 1] \geq \Pr[Y = 1]$, while the analogue of moving in the x direction is performed otherwise. In order for our limit protocol to behave identical to the buzzer protocol, we would like the region $\Pr[X = 1] \geq \Pr[Y = 1]$ to correspond to the region $p \geq q$. This is done by using a symmetric reference distribution.

Next, we would like to show a stability result, proving that every protocol performing the task $[\mathrm{AND}, 0]$ with nearly optimal information complexity is similar to the buzzer protocol. We measure similarity in terms of the leaf distribution $(\mathbf{p}, \mathbf{q})$, and define the following potential function:

▶ **Definition 29.** Given a protocol π for $[\mathrm{AND}, 0]$, a constant $0 < c < 1$, and a pretend distribution μ, let

$$\Phi_{c,\mu}(\pi) = \mathbb{E}\left[((c - \boldsymbol{\ell})_+)^2\right],$$

where $(\cdot)_+ = \max\{\cdot, 0\}$, and $\boldsymbol{\ell} = \max(\mathbf{p}, \mathbf{q})$. Denote $\Phi_{c,\mu} = \Phi_{c,\mu}(\pi^*)$, where π^* is the buzzer protocol.

The following theorem shows that the value of the potential function is small for nearly optimal protocols.

▶ **Theorem 30.** *Let $\overline{\mu}$ be a full support distribution, and $\overline{\mu} = \nu \odot \mu$ be its decomposition, where ν is a symmetric reference distribution and $\mu = (p, q)$ is the product pretend distribution. Assume that $c \leq \max\{p, q\}$. Let π be a protocol performing $[\mathrm{AND}, 0]$. Then*

$$\Phi_{c,\mu}(\pi) = O(\mathrm{IC}_{\overline{\mu}}(\pi) - \mathrm{IC}_{\overline{\mu}}(\mathrm{AND}, 0)) = O(\mathrm{IW}_{\overline{\mu}}(\pi)).$$

The constant in the $O(\cdot)$ is uniform whenever $\nu(0,0), \nu(0,1), \nu(1,0), p, q$ are bounded away from 0 and 1.

In order to prove this theorem, we measure how each performed step contributes both to the wastage and to the potential function. To measure the wastage, we work with SIM instead of IC, as it is a more natural measure for this task.

▶ **Lemma 31.** *Let $\overline{\mu}$ be a full support distribution, and $\overline{\mu} = \nu \odot \mu$ be its decomposition, where ν is a symmetric reference distribution and μ is the pretend distribution. Let $0 < c < 1$, and let π be the protocol which behaves as follows:*

1. *One step of a pretend random walk is performed, which corresponds to one bit that is sent in the protocol.*
2. *The pretend random walk that corresponds to the buzzer protocol is simulated from that point: assuming that after the first bit was sent the pretend distribution is (p, q), let $\pi^*_{(p,q)}$ be the buzzer protocol for the input distribution $\nu \odot (p, q)$. Then, the pretend random walk that corresponds to $\pi^*_{(p,q)}$ is simulated (the value of (p, q) is different for the case that the first bit equals 1, and when it equals 0).*

Then

$$\Phi_{c,\mu}(\pi) - \Phi_{c,\mu} = O_\nu(\mathrm{SIM}_\mu(\mathrm{AND}, 0) - \mathrm{SIM}_\mu(\pi)).$$

The constant in the $O(\cdot)$ is uniform whenever $\nu(0,0), \nu(0,1), \nu(1,0), c$ are bounded away from 0 and 1.

The potential function of Definition 29 is defined in that manner so that Lemma 31 holds. Let us elaborate on this: assume that a protocol π is defined as in this lemma, with a pretend input distribution of (p, q). Assume that the first step moves from (p, q) either to $(p + \delta)$ or to $(p - \delta)$ with equal probability. Then

$$\begin{aligned}\mathrm{SIM}_{(p,q)}(\pi) - \mathrm{SIM}_{(p,q)}(\mathrm{AND}, 0) &= \frac{1}{2}\mathrm{SIM}_{(p+\delta,q)}(\mathrm{AND}, 0) \\ &\quad + \frac{1}{2}\mathrm{SIM}_{(p+\delta,q)}(\mathrm{AND}, 0) - \mathrm{SIM}_{(p,q)}(\mathrm{AND}, 0) \\ &\approx \frac{\delta^2}{2}\frac{\partial^2}{\partial p^2}\mathrm{SIM}_{(p,q)}(\mathrm{AND}, 0).\end{aligned}$$

Thus, this difference has the same order of magnitude as δ^2. We would like the change in the potential function to have the same order. Looking at the function x^2, it holds that

$$\frac{1}{2}(x + \delta)^2 + \frac{1}{2}(x - \delta)^2 - x^2 = \frac{\delta^2}{2}.$$

If a protocol π moves according to the direction of the buzzer protocol, then π is the same as π^* and both differences are zero. Therefore, assume that $p > q$, and π moves in the x direction, whereas the buzzer protocol would have moved in the y direction. Roughly speaking, the leaf distribution of π is obtained from the leaf distribution of π^* by splitting some of the mass around $\boldsymbol{\ell} \approx p$ between $\boldsymbol{\ell} \approx p - \delta$ and $\boldsymbol{\ell} \approx p + \delta$. Thus, $\Phi_{c,\mu}(\pi) - \Phi_{c,\mu}$ approximately has the order of magnitude of

$$\frac{1}{2}(c - p - \delta)^2 + \frac{1}{2}(c - p + \delta)^2 - (c - p)^2 = \frac{\delta^2}{2}.$$

We chose $(c - p)^2_+$ instead of $(c - p)^2$ since Lemma 36 requires the buzzer protocol to have a value of zero. Indeed, by choosing c carefully we can achieve this.

We will prove Lemma 31 using the following criterion.

▶ **Lemma 32.** *Let ν be a symmetric reference distribution, and $C > 0$ a constant. Define $F(p,q) = C\,\mathrm{SIM}_{(p,q)}(\mathrm{AND}, 0) + \Phi_{c,(p,q)}$. If for every q, $F(p,q)$ is concave as a function of p, and for every p, $F(p,q)$ is concave as a function of q, then Lemma 31 holds, and the constant in the $O(\cdot)$ can be taken to be equal to C.*

Proof. Let π be the protocol defined in Lemma 31, and let μ be its pretend input distribution. Assume that the pretend random walk of π first moves from μ either to μ_0 or to μ_1, with probabilities λ_0 and λ_1. We assume this step is on the x-direction, thus, the first step is from (p,q) to (p_0,q) or (p_1,q). The analysis for the case that this step it in the y-direction is similar. Let $0 < c < 1$. Then $\mathrm{SIM}_{(p,q)}(\pi) = \sum_b \lambda_b \mathrm{SIM}_{(p_b,q)}(\mathrm{AND}, 0)$, and $\Phi_{c,(p,q)} = \sum_b \lambda_b \Phi_{c,(p_b,q)}$. From concavity,

$$\begin{aligned} C\,\mathrm{SIM}_{(p,q)}(\mathrm{AND},0) + \Phi_{c,(p,q)} &= F(p,q) \geq \sum_b \lambda_b F(p_b,q) \\ &= \sum_b \lambda_b (C\,\mathrm{SIM}_{(p_b,q)}(\mathrm{AND},0) + \Phi_{c,(p_b,q)}) \\ &= C\,\mathrm{SIM}_{(p,q)}(\pi) + \Phi_{c,(p,q)}(\pi). \end{aligned}$$

◀

Thus, our focus would be proving that these concavity conditions hold for some value C. We proceed by calculating $\Phi_{c,(p,q)}$, assuming without loss of generality that $p \geq q$. One can see that whenever $p \geq c$, with probability 1 the leaf distribution of the buzzer protocol satisfies $\boldsymbol{\ell} \geq p \geq c$, and thus the potential function evaluates to 0. Consider the case $p < c$. Using the leaf distribution, we obtain the formula

$$\Phi_{c,(p,q)} = (1 - q/p)(c-p)^2 + 2\int_{\ell=p}^{c} \frac{pq}{\ell^3}(c-\ell)^2 d\ell.$$

Thus, the general definition is as follows:

$$\Phi_{c,(p,q)} = \begin{cases} 0 & \text{if } \max\{p,q\} \geq c, \\ (1-q/p)(c-p)^2 + 2\int_{\ell=p}^{c} \frac{pq}{\ell^3}(c-\ell)^2 d\ell & \text{if } q \leq p \leq c, \\ (1-p/q)(c-q)^2 + 2\int_{\ell=q}^{c} \frac{pq}{\ell^3}(c-\ell)^2 d\ell & \text{if } p \leq q \leq c. \end{cases}$$

In order to apply Lemma 32, we start by showing that the function $\Phi_{c,(p,q)}$ is differentiable for all p (in the direction of p) given a fixed value of q, and for all q given a fixed value of p. This is done by calculating the two one-sided derivatives in the points suspected of non-differentiability: $p = q$ and $\max\{p,q\} = c$. To state it into more detail, for any fixed q, we calculate both

$$\left.\frac{\partial \Phi_{c,(p,q)}}{\partial p}\right|_{+} = \lim_{h \to 0^+} \frac{\Phi_{c,(p+h,q)} - \Phi_{c,(p,q)}}{h},$$

and

$$\left.\frac{\partial \Phi_{c,(p,q)}}{\partial p}\right|_{-} = \lim_{h \to 0^-} \frac{\Phi_{c,(p+h,q)} - \Phi_{c,(p,q)}}{h},$$

and verify that both values are equal in all suspected points. We do the same switching the roles of p and q. (though it is not required as this potential function is symmetric, since we assume the reference distribution to be symmetric) Additionally, we calculate its second derivatives whenever they are defined. If $\max\{p,q\} > c$, then they are trivially zero. For $q < p < c$, we get:

$$\frac{\partial^2 \Phi_{c,(p,q)}}{\partial p^2} = 2(1 - q/p)$$

and

$$\frac{\partial^2 \Phi_{c,(p,q)}}{\partial q^2} = 0.$$

Actually, there is a reason why this second derivative with respect to q is zero. For any $0 < \delta \leq \min\{p-q, q\}$, consider a protocol π that first moves to $(p, q-\delta)$ or to $(p, q+\delta)$, each with probability $1/2$, and then simulates the buzzer protocol. It has the same leaf distribution as the buzzer protocol (in the pretend world). Both the buzzer protocol and π either get to the point $(p, 0)$ or to the point (p, p), with probabilities $1 - q/p$ and q/p, respectively. From that point on, both continue the same way, resulting in the same leaf distribution. This validates the equality

$$\Phi_{c,(p,q)} = \frac{1}{2}\Phi_{c,(p,q+\delta)} + \frac{1}{2}\Phi_{c,(p,q-\delta)}$$

for all q and δ sufficiently small, which implies linearity in the region $q \in [0, p]$ (given a fixed p).

Similar calculations will now be performed with regard to $\mathrm{SIM}_{p,q}(\mathrm{AND}, 0)$. Denote $x = \nu(0,0), y = \nu(1,0) = \nu(0,1), z = \nu(1,1)$. It is possible to extract the value of this function from the equations in [4], using the conversion from SIM to CI (46) and from CI to IC (9). Nevertheless, we calculate it using the formula (49), which is an expectation over a value obtained in the leafs of the protocol. Let $p \geq q$, and let Π correspond to the buzzer protocol, which starts at distribution (p, q). Then,

$$\begin{aligned}
\mathrm{SIM}_{p,q}(\mathrm{AND}, 0) &= \mathop{\mathbb{E}}_{\Pi}[\langle \nu, \mu_\Pi \rangle (H_{\mu_\Pi}(X|Y) + H_{\mu_\Pi}(Y|X))] \\
&= \left(1 - \frac{q}{p}\right)((1-p)x + py)h\left(\frac{py}{(1-p)x+py}\right) + \\
&\quad \int_p^1 \frac{2pq}{\ell^3}((1-\ell)x + \ell y)h\left(\frac{y\ell}{x(1-\ell)+y\ell}\right)\mathrm{d}\ell \\
&= -\left[q(1-p)y + (1-p)(1-q)x\log\frac{(1-p)x}{(1-p)x+py} + \right. \\
&\quad \left.\left(\frac{pqy^2}{x} + (p+q-2pq)y\right)\log\frac{py}{(1-p)x+py}\right].
\end{aligned}$$

Calculating the second derivative, we get for $p > q$,

$$\frac{\partial^2 \mathrm{SIM}_{(p,q)}(\mathrm{AND}, 0)}{\partial p^2} = -2(1-q/p)\frac{xy}{2(1-p)p^2((1-p)x+py)},$$

and

$$\frac{\partial^2 \mathrm{SIM}_{(p,q)}(\mathrm{AND}, 0)}{\partial q^2} = 0.$$

The reason that the second derivative is zero is the same as explained for the potential function. For proving differentiability (on each direction separately), the only suspected point is $p = q$. Comparing the two one-sided derivatives implies the result.

Now we are almost ready to apply Lemma 32. Define

$$C = \max_{0 \leq p \leq 1} \frac{2(1-p)p^2((1-p)x+py)}{xy},$$

and $F(p,q) = C\,\mathrm{SIM}_\mu(\pi^*) + \Phi_{c,\mu}$. For any fixed q, $\frac{\partial F(p,q)}{\partial p}$ is continuous, piecewise differentiable, and its derivative, $\frac{\partial}{\partial p}\frac{\partial F(p,q)}{\partial p}$ is non-positive wherever it is defined. Thus, $\frac{\partial F(p,q)}{\partial p}$ is non-increasing, and $F(p,q)$ is concave as a function of p. The same holds when switching the roles of p and q, thus the conditions in Lemma 32 are satisfied, which concludes the proof of Lemma 31. Finally, we are able to prove Theorem 30.

Proof of Theorem 30. Let T be the protocol tree of π. This is a directed binary tree with two children for each internal node. Each node corresponds to a state of the protocol when some communication has taken place, and its children are the two consecutive states, chosen according to the bit sent by the player owning the node.

We can construct T using a sequence of trees, $T_1, T_2, \ldots, T_k = T$. The tree T_1 contains only the root of T, and for all i, T_i is obtained from T_{i-1} by adding the children of a leaf of T_{i-1} which is not a leaf of T.

Given a tree T_i, construct a protocol π_i, that whenever it reaches a state represented by node v which is not a leaf of T_i, the protocol behaves as π for the next bit sent, and if the state is represented by a leaf of T_i, then the buzzer protocol is simulated from that point on. Let D be the constant in the $O(\cdot)$ guaranteed from Lemma 31. The lemma implies that for all i, $\Phi_{c,\mu}(\pi_i) - \Phi_{c,\mu}(\pi_{i-1}) \le D(\mathrm{SIM}_\mu(\pi_{i-1}) - \mathrm{SIM}_\mu(\pi_i))$. Summing over i, we get a telescopic summation that results in

$$\Phi_{c,\mu}(\pi) = \Phi_{c,\mu}(\pi_k) - \Phi_{c,\mu}(\pi_1) \le D(\mathrm{SIM}_\mu(\pi_1) - \mathrm{SIM}_\mu(\pi_k)) = D(\mathrm{SIM}_\mu(\mathrm{AND},0) - \mathrm{SIM}_\mu(\pi)).$$

We used the fact that $\Phi_{c,\mu}(\pi_1) = \Phi_{c,\mu} = 0$, which hold since we assumed that $c \le \max\{p,q\}$, and the leaf distribution of the buzzer protocol has zero mass on $\ell < \max\{p,q\}$, therefore its potential cost is zero. This finishes the proof as

$$\mathrm{SIM}_\mu(\mathrm{AND},0) - \mathrm{SIM}_\mu(\pi) = \langle \nu,\mu\rangle(\mathrm{CI}_{\overline{\mu}}(\mathrm{AND},0) - \mathrm{CI}_{\overline{\mu}}(\pi)) = \langle \nu,\mu\rangle\,\mathrm{IW}_{\overline{\mu}}(\pi) \le \mathrm{IW}_{\overline{\mu}}(\pi). \quad \blacktriangleleft$$

6.2 Lower bound on the information complexity of $\mathrm{IC}_\mu(\mathrm{AND},\epsilon)$

In this section, we prove Theorem 9 by showing that every distribution $\overline{\mu}$ which is of full support, except perhaps for $\overline{\mu}(1,1)$, satisfies $\mathrm{IC}_{\overline{\mu}}(\mathrm{AND},\epsilon) \ge \mathrm{IC}_{\overline{\mu}}(\mathrm{AND},0) - O(\overline{h}(\epsilon))$. Recall that Theorem 9(ii) follows from Part (i) and we have already established the upper bound of Theorem 9(i) in Theorem 5. Hence it remains to prove the following theorem.

▶ **Theorem 33** (The remaining case of Theorem 9)**.** *Let $\overline{\mu}$ be a full-support distribution, except perhaps for $\overline{\mu}(1,1)$. For all $\epsilon \ge 0$,*

$$\mathrm{IC}_{\overline{\mu}}(\mathrm{AND},\epsilon) \ge \mathrm{IC}_{\overline{\mu}}(\mathrm{AND},0) - O_{\overline{\mu}}(\overline{h}(\epsilon)).$$

The hidden constant can be fixed if $\overline{\mu}(0,0), \overline{\mu}(0,1), \overline{\mu}(1,0)$ are bounded away from 0.

The proof uses the idea of *protocol completion*: given a protocol π performing $[\mathrm{AND},\epsilon]$, we can create a protocol π_0, which we call the zero-error *completion* of π. Such a protocol π_0 takes the following steps:

- First Alice and Bob simulate π until it terminates.
- Afterwards they run a protocol that solves the AND function with zero error.

The *cost of completion* is the amount of information revealed in the second step, and it is equal to $\mathrm{IC}_{\overline{\mu}}(\pi_0) - \mathrm{IC}_{\overline{\mu}}(\pi)$. We have shown in the proof of Theorem 7 that for general functions, this cost is bounded by $O(\overline{h}(\sqrt{\epsilon}))$, but here we would like to prove a stronger bound of $O(\overline{h}(\epsilon))$ for protocols that are almost optimal for the AND function. This obviously

would yield the desired lower bound, and prove Theorem 33. This completion cost can be arbitrarily close to $\mathbb{E}_\Pi[\mathrm{IC}_{\overline{\mu}_\Pi}(\mathrm{AND}, 0)]$. In order to bound this quantity, we first bound the information complexity of the AND function.

▶ **Lemma 34.** *Consider a reference distribution ν with $\nu(0,0) = x, \nu(1,0) = \nu(0,1) = y, \nu(1,1) = z$, such that $x, y, z > 0$. Let $\mu = (p, q)$ be a pretend distribution. Let $\overline{\mu} = \nu \odot \mu$, and $\overline{\mu}(1,1) = \delta$. Let $0 < C < 1$ be an arbitrary constant.*

Firstly $\mathrm{IC}_{\overline{\mu}}(\mathrm{AND}, 0) \leq 2\overline{h}(1-\delta)$. *Secondly*

$$\mathrm{IC}_{\overline{\mu}}(\mathrm{AND}, 0) \leq \begin{cases} O(\overline{h}(\delta/z)) & \text{if } \max(p,q) \geq C, \\ O(\overline{h}(\sqrt{\delta/z})) & \text{if } p, q < C. \end{cases}$$

The hidden constants can be fixed if x, y, C are bounded away from both 0 and 1.

Proof. First we prove that $\mathrm{IC}_{\overline{\mu}}(\mathrm{AND}, 0) \leq 2\overline{h}(1-\delta)$. Assume that $\delta \geq 1/2$, as otherwise the inequality trivially follows. The information complexity is achieved by a protocol where both Alice and Bob send their inputs. The cost of that protocol is at most $H(XY) \leq H(X) + H(Y) \leq 2h(\delta)$.

For proving the other bounds, assume that $\delta < 1/2$, since otherwise the lemma trivially follows. If $p, q > 1/2$, then $\delta = \frac{\nu(1,1)pq}{\langle \nu, \mu \rangle} \geq \nu(1,1) = z$, as

$$\langle \nu, \mu \rangle = (1-p)(1-q)x + [p(1-q) + (1-p)q]y + pqz \leq (x + 2y + z)pq = pq.$$

In this case, the lemma follows.

Assume that either $p \leq 1/2$ or $q \leq 1/2$. Without loss of generality, $p \leq q$. We will analyze the protocol in which Alice first sends her input to Bob, and if $X = 1$ then Bob sends his input to Alice. This protocol has a cost of

$$H(X|Y) + \Pr[X=1]H(Y|X=1) \leq H(X) + \Pr[X=1] \leq \overline{h}(\Pr[X=1]) + \Pr[X=1].$$

The obtained bound is monotonic in $\Pr[X = 1]$, a fact that we will use.

Now

$$\Pr[X=1] = \frac{p(1-q)y + pqz}{\langle \nu, \mu \rangle} \leq \frac{p(y+z)}{\langle \nu, \mu \rangle} = \frac{\delta(y+z)}{zq}.$$

Thus, if $q \geq C$, then the cost of completion is at most

$$\overline{h}\left(\frac{\delta(y+z)}{zC}\right) + \frac{\delta(y+z)}{zC} \leq \frac{(y+z)\delta}{Cz} + \begin{cases} \overline{h}(\delta/z) & \text{if } \frac{y+z}{C} < 1, \\ \frac{y+z}{C} 2\overline{h}(\delta/z) & \text{otherwise,} \end{cases} \tag{50}$$

using the bound $\overline{h}(cx) \leq 2c\overline{h}(x)$ for all $c > 1$, from (12).

If $q \leq C$, $\Pr[X = 1]$ is maximized at $q = p$. Assume indeed that $p = q$. We will bound its value from below. The equation $\frac{q^2 z}{\langle \nu, \mu \rangle} = \frac{q^2 z}{\langle \nu, \mu \rangle} = \delta$ implies

$$q = \sqrt{\frac{\delta \langle \nu, \mu \rangle}{z}}.$$

Now since

$$\langle \nu, \mu \rangle \geq \nu(0,0)\mu(0,0) = (1-p)(1-q)x \geq (1-C)^2 x,$$

we have

$$\Pr[X=1] \leq \frac{\delta(y+z)}{zq} \leq \sqrt{\frac{\delta}{z}} \frac{y+z}{(1-C)\sqrt{x}}.$$

The proof concludes applying similar calculations as in (50). ◀

Next, we use this bound to show that if the probability that $\max\{\mathbf{p}, \mathbf{q}\}$ does not exceed some constant is very small, then one can get an improvement over $\overline{h}(\sqrt{\epsilon})$ for the completion cost.

▶ **Lemma 35.** *Let ν be a symmetric reference distribution with $\nu(0,0) = x$, $\nu(0,1) = \nu(1,0) = y$ and $\nu(1,1) = z > 0$. Let $\mu = (p,q)$ be a pretend distribution, and let $\overline{\mu} = \nu \odot \mu = \nu$.*

Let π be a protocol performing $[\mathrm{AND}, \epsilon]$. Let $0 < C < 1$ be an arbitrary constant, $\kappa = \Pr[\max\{\mathbf{p}, \mathbf{q}\} \le C]$.

The protocol π can be completed to a zero-error protocol using an additional information cost of

$$O\left(\kappa \overline{h}(\sqrt{\epsilon/\kappa}) + (1-\kappa)\overline{h}(\tfrac{\epsilon}{1-\kappa})\right),$$

where the cost is according to the distribution $\overline{\mu}$, and the hidden constant in $O(\cdot)$ can be fixed if x, y, p, q, C are all bounded away from both 0 and 1.

Proof. First, note that

$$\overline{\mu}(1,1) = \frac{zpq}{\langle \nu, \mu \rangle} \le \frac{zpq}{x(1-p)(1-q)} = O(z).$$

Let $\boldsymbol{\psi}$ be the random variable denoting the completion cost as a function of Π. Let $\mathbf{1}_{o=b}$ be the indicator of whether π outputs b given the transcript Π, for $b = 0, 1$. The total completion cost is

$$\mathbb{E}[\boldsymbol{\psi}] = \sum_{b=0,1} \mathbb{E}[\boldsymbol{\psi}\mathbf{1}_{o=b}].$$

We start by bounding $\mathbb{E}[\boldsymbol{\psi}\mathbf{1}_{o=1}]$. Let $\boldsymbol{\delta}$ be the random variable which equals $\overline{\mu}_{\Pi}(1,1)$.

$$\mathbb{E}[(1-\boldsymbol{\delta})\mathbf{1}_{o=1}] = \Pr[(X,Y) \ne (1,1), \pi \text{ outputs } 1] \le \epsilon.$$

From Lemma 34, the completion cost $\boldsymbol{\psi}$ is at most $2\overline{h}(1-\boldsymbol{\delta})$. From the concavity of $\overline{h}$,

$$\mathbb{E}[\boldsymbol{\psi}\mathbf{1}_{o=1}] = \mathbb{E}\, O(\overline{h}(1-\boldsymbol{\delta}))\mathbf{1}_{o=1} = \mathbb{E}\, O(\overline{h}((1-\boldsymbol{\delta})\mathbf{1}_{o=1})) \le O(\overline{h}(\mathbb{E}[(1-\boldsymbol{\delta})\mathbf{1}_{o=1}])) \le O(\overline{h}(\epsilon)).$$

This can be bounded as desired since in both cases of $\kappa > 1/2$ and $\kappa \le 1/2$, we have

$$\overline{h}(\epsilon) = O\left(\kappa \overline{h}(\sqrt{\epsilon/\kappa}) + (1-\kappa)\overline{h}(\tfrac{\epsilon}{1-\kappa})\right).$$

Next we bound $\mathbb{E}[\boldsymbol{\psi}\mathbf{1}_{o=0}]$.

$$\mathbb{E}[\boldsymbol{\delta}\mathbf{1}_{o=0}] = \Pr[(X,Y) = (1,1), \pi \text{ outputs } 0] \le \epsilon\overline{\mu}(1,1) \le \epsilon O(z).$$

Let S be the event that $\max\{\mathbf{p}, \mathbf{q}\} \le C$. Then,

$$\mathbb{E}[\boldsymbol{\delta}\mathbf{1}_{o=0}|S] \le \epsilon O(z)/\Pr[S] = \epsilon O(z)/\kappa.$$

$$\mathbb{E}[\boldsymbol{\delta}\mathbf{1}_{o=0}|\overline{S}] \le \epsilon O(z)/(1-\kappa).$$

From Lemma 34, the completion cost is of order of $\overline{h}\left(\sqrt{\boldsymbol{\delta}/z}\right)$ when S happens, and $\overline{h}(\boldsymbol{\delta}/z)$ otherwise.

$$\begin{aligned}
\mathbb{E}[\boldsymbol{\psi}\mathbf{1}_{o=0}] &= \Pr[S]\,\mathbb{E}[\boldsymbol{\psi}\mathbf{1}_{o=0}|S] + \Pr[\overline{S}]\,\mathbb{E}[\boldsymbol{\psi}\mathbf{1}_{o=0}|\overline{S}] \\
&= O\left(\kappa\,\mathbb{E}\left[\overline{h}\left(\sqrt{\boldsymbol{\delta}\mathbf{1}_{o=0}/z}\right)|S\right] + (1-\kappa)\,\mathbb{E}[\overline{h}(\boldsymbol{\delta}\mathbf{1}_{o=0}/z)|\overline{S}]\right) \\
&\le O\left(\kappa\overline{h}\left(\sqrt{\mathbb{E}[\boldsymbol{\delta}\mathbf{1}_{o=0}|S]/z}\right) + (1-\kappa)\overline{h}(\mathbb{E}[\boldsymbol{\delta}\mathbf{1}_{o=0}|\overline{S}]/z)\right) \qquad (51) \\
&\le O\left(\kappa\overline{h}\left(\sqrt{O(\epsilon)/\kappa}\right) + (1-\kappa)\overline{h}(O(\epsilon)/(1-\kappa))\right) \\
&\le O\left(\kappa\overline{h}\left(\sqrt{\epsilon/\kappa}\right) + (1-\kappa)\overline{h}\left(\frac{\epsilon}{1-\kappa}\right)\right), \qquad (52)
\end{aligned}$$

where (51) follows from the concavity of $\overline{h}(\cdot/z)$ and $\overline{h}(\sqrt{\cdot/z})$, and (52) follows from (12). ◀

Consider an almost optimal protocol π_0 so that $\mathrm{IC}_{\overline{\mu}}(\pi_0) - \mathrm{IC}_{\overline{\mu}}(\mathrm{AND}, 0)$ is small. Our stability result, Theorem 30, translates this to a bound on the potential function introduced in Definition 29. The next lemma uses this to show that for such a protocol π_0, one can obtain a strong bound on the value of κ in Lemma 35.

▶ **Lemma 36.** *Let $\overline{\mu}$ be full-support distribution and let $\overline{\mu} = \nu \odot \mu$ be its decomposition, where ν is a symmetric reference distribution, and μ is the pretend distribution. Let $c = \max\{\Pr_\mu[X=1], \Pr_\mu[Y=1]\}$. Let π be an arbitrary protocol, and π_0 be the completion of π to a protocol performing* $[\mathrm{AND}, 0]$*. Then*

$$\Pr[\max\{\mathbf{p}, \mathbf{q}\} \le \frac{c}{4}] = O_{c,\mu,\nu}(\mathrm{IC}_{\overline{\mu}}(\pi_0) - \mathrm{IC}_{\overline{\mu}}(\mathrm{AND}, 0)),$$

The hidden constant can be fixed if $p, q, \mu(0,0), \mu(0,1), \mu(1,0)$ are all bounded away from both 0 *and* 1*, where $\mu = (p, q)$.*

Proof. Let $\boldsymbol{\ell}_{p,q}$ be the distribution of $\boldsymbol{\ell}$ that corresponds to the buzzer protocol when it is invoked from a pretend distribution parametrized by (p, q).

We start by showing that for any $0 < p, q < 1$,

$$\Pr[\boldsymbol{\ell}_{p,q} \le 2\max\{p, q\}] \ge \frac{3}{4}.$$

Assume without loss of generality that $p \ge q$. Using the leaf distribution from Section 6.1,

$$\Pr[p \le \boldsymbol{\ell} \le 2p] = 2\int_p^{2p} \frac{pq}{\ell^3}\,\mathrm{d}\ell + \left(1 - \frac{q}{p}\right) > \frac{3}{4}.$$

This implies

$$\begin{aligned}
\Pr[\boldsymbol{\ell}_{\pi_0} \le \frac{c}{2}] &= \Pr\left[\boldsymbol{\ell}_{\pi_0} \le 2\frac{c}{4}\right] \\
&\ge \Pr\left[\max\{\mathbf{p}, \mathbf{q}\} \le \frac{c}{4}\right] \Pr\left[\ell_{\mathbf{p},\mathbf{q}} \le 2\max\{\mathbf{p}, \mathbf{q}\}\right] \\
&\ge \frac{3}{4}\Pr\left[\max\{\mathbf{p}, \mathbf{q}\} \le \frac{c}{4}\right].
\end{aligned}$$

Markov's inequality and Theorem 30 imply

$$\begin{aligned}
\Pr[\boldsymbol{\ell}_{\pi_0} \le \frac{c}{2}] &= \Pr[(c - \boldsymbol{\ell}_{\pi_0})_+^2 \ge \frac{c^2}{4}] \le \frac{\mathbb{E}[(c - \boldsymbol{\ell}_{\pi_0})_+^2]}{c^2/4} = \frac{\Phi_{c,\mu}(\pi_0)}{c^2/4} \\
&= O(\mathrm{IC}_{\overline{\mu}}(\pi_0) - \mathrm{IC}(\mathrm{AND}, 0)).
\end{aligned}$$

◀

Now we are ready to prove Theorem 33, and thus complete the proof of Theorem 9.

Proof of Theorem 33. We first prove the theorem for the full-support distributions. Consider such a distribution $\overline{\mu}$. Let π be a protocol performing $[\mathrm{AND}, \epsilon]$. We can assume that $\mathrm{IC}_{\overline{\mu}}(\pi) \leq \mathrm{IC}_{\overline{\mu}}(\mathrm{AND}, 0)$, and let $C = \max\{\Pr_\mu[X=1], \Pr_\mu[Y=1]\}/4$, $\kappa = \Pr[\max\{\mathbf{p}, \mathbf{q}\} \leq C]$. Lemma 35 constructs a zero-error protocol π_0 whose wastage w is at most

$$w = O\left(\kappa \overline{h}\left(\sqrt{\frac{\epsilon}{\kappa}}\right) + (1-\kappa)\overline{h}\left(\frac{\epsilon}{1-\kappa}\right)\right).$$

Lemma 36 states that $\kappa = O(w)$, and so

$$\kappa = O\left(\kappa \overline{h}\left(\sqrt{\frac{\epsilon}{\kappa}}\right) + (1-\kappa)\overline{h}\left(\frac{\epsilon}{1-\kappa}\right)\right).$$

If $\frac{\epsilon}{1-\kappa} \leq 1/2$, then (12) shows that

$$\kappa = O\left(\kappa \overline{h}\left(\sqrt{\frac{\epsilon}{\kappa}}\right) + \overline{h}(\epsilon)\right). \tag{53}$$

Otherwise, $\kappa \geq 1 - 2\epsilon \geq 1/2$ (assuming $\epsilon \leq 1/4$), and so

$$\kappa = O(h(\sqrt{\epsilon}) + (1-\kappa)) = O(h(\sqrt{\epsilon}) + \epsilon),$$

which contradicts $\kappa \geq 1/2$ for small enough ϵ.

Denoting the hidden constant in (53) by M, we get

$$\left(1 - Mh\left(\sqrt{\frac{\epsilon}{\kappa}}\right)\right)\kappa \leq Mh(\epsilon).$$

We will show that for small ϵ, this forces $\kappa \leq 2Mh(\epsilon)$. Indeed, suppose that $\kappa > 2Mh(\epsilon)$, which implies that $\kappa > 2M\epsilon\log(1/\epsilon)$. Then

$$\frac{\epsilon}{\kappa} < \frac{1}{2M\log(1/\epsilon)},$$

and so for small enough ϵ, $Mh(\sqrt{\epsilon/\kappa}) < 1/2$. This shows that

$$\left(1 - Mh\left(\sqrt{\frac{\epsilon}{\kappa}}\right)\right)\kappa > \frac{\kappa}{2} > Mh(\epsilon),$$

contradicting the inequality above. We conclude that for small ϵ we have $\kappa = O(h(\epsilon))$.

Applying Lemma 35 again, we see that

$$\mathrm{IC}_{\overline{\mu}}(\pi_0) - \mathrm{IC}_{\overline{\mu}}(\pi) \leq \kappa O\left(\overline{h}\left(\sqrt{\frac{\epsilon}{\kappa}}\right)\right) + O(\overline{h}(\epsilon)) \leq O(\kappa) + O(\overline{h}(\epsilon)) = O(\overline{h}(\epsilon)).$$

Since $\mathrm{IC}_{\overline{\mu}}(\pi_0) \geq \mathrm{IC}_{\overline{\mu}}(\mathrm{AND}, 0)$, we conclude that $\mathrm{IC}_{\overline{\mu}}(\pi) \geq \mathrm{IC}_{\overline{\mu}} - O(\overline{h}(\epsilon))$.

Next consider a distribution $\overline{\mu}$ with $\overline{\mu}(1,1) = 0$, that assigns a strictly positive probability for every other input. There is a series of full support distributions, $\overline{\mu}_1, \overline{\mu}_2, \ldots$ that converge to $\overline{\mu}$, and assume without loss of generality that for every input $a \in \{0,1\}^2$ and for every $n \in \mathbb{N}$, $\overline{\mu}_n(a) \geq \overline{\mu}(a)/2$. From the continuity of information complexity with respect to the tasks $[\mathrm{AND}, 0]$ and $[\mathrm{AND}, \epsilon]$,

$$\lim_{n\to\infty} \mathrm{IC}_{\overline{\mu}_n}(\mathrm{AND}, 0) = \mathrm{IC}_{\overline{\mu}}(\mathrm{AND}, 0),$$

and

$$\lim_{n\to\infty} \mathrm{IC}_{\overline{\mu}_n}(\mathrm{AND}, 0) = \mathrm{IC}_{\overline{\mu}}(\mathrm{AND}, 0).$$

Assume that $\overline{\mu}(0,0), \overline{\mu}(0,1), \overline{\mu}(1,0)$ are bounded from below. It is possible to decompose $\overline{\mu}$ into $\nu \odot (p, q)$, where ν is symmetric and $p, q, \nu(0,0), \nu(0,1)$ and $\nu(1,0)$ are bounded. This is done by considering a decomposition where $p = 1/2$ and q is chosen such that ν is symmetric. Therefore, there is a constant $C > 0$ such that

$$\mathrm{IC}_{\mu_n}(\mathrm{AND}, \epsilon) \geq \mathrm{IC}_{\mu_n}(\mathrm{AND}, \epsilon) - C\overline{h}(\epsilon).$$

Thus,

$$\mathrm{IC}_{\mu}(\mathrm{AND}, \epsilon) \geq \mathrm{IC}_{\mu}(\mathrm{AND}, \epsilon) - C\overline{h}(\epsilon). \qquad \blacktriangleleft$$

7 The set disjointness function with error

In this section we present the proofs of the results concerning the set disjointness function. It will be convenient to switch the roles of 0 and 1 in the range of the function, and redefine DISJ_n as $\mathrm{DISJ}_n(X, Y) = \vee_{i=1}^n (X_i \wedge Y_i)$, i.e. $\mathrm{DISJ}_n(X, Y) = 0$ if the inputs are disjoint and it is equal to 1 otherwise. Obviously, this will not affect the correctness of our results.

7.1 Proof of Theorem 13

Theorem 13 (restated). *For the set disjointness function* DISJ_n *on inputs of length* n*, we have*

$$R_\epsilon(\mathrm{DISJ}_n) = n[\mathrm{IC}^0(\mathrm{AND}, 0) - \Theta(h(\epsilon))].$$

As discussed in Section 3.4, we only need to prove the upper bound. In fact, we will prove the following lemma, from which Theorem 13 follows using Corollary 10.

▶ **Lemma 37.** *For every* $\epsilon > 0$ *and sufficiently large* n*,*

$$\frac{R_\epsilon(\mathrm{DISJ}_n)}{n} \leq \mathrm{IC}^0(\mathrm{AND}, \epsilon, 1 \to 0) + o_{n\to\infty}(1).$$

Intuitively, an upper bound like Lemma 37 is essentially a compression result. Besides, as DISJ_n has a self-reducible structure (see [5]), one can make use of this fact together with the Braverman–Rao [6] compression. A difficulty is that what we want to solve is $[\mathrm{DISJ}_n, \epsilon]$, that is, the error allowed is non-distributional, while the error unavoidably introduced in the compression phase is distributional. Fortunately, this can be salvaged by a minimax argument introduced in Section 6.2 of [3].

In order to use self-reducibility and compression, one first needs to have a control on the information cost of solving $[\mathrm{DISJ}_n, \epsilon]$.

▶ **Lemma 38.** *For every* $\epsilon > 0$ *and sufficiently large* n*,*

$$\mathrm{IC}(\mathrm{DISJ}_n, \epsilon, 1 \to 0) \leq n\,\mathrm{IC}^0(\mathrm{AND}, \epsilon, 1 \to 0) + o(n),$$

where $\mathrm{IC}(\mathrm{DISJ}_n, \epsilon, 1 \to 0) := \max_\mu \mathrm{IC}_\mu(\mathrm{DISJ}_n, \epsilon, 1 \to 0)$.

The proof is a direct adaptation of the proof for Lemma 8.5 in [4].

Proof. Let Ω_0 denote the set of all measures μ on $\{0,1\}^2$ with $\mu(1,1) = 0$. Let π be a protocol that computes $[\mathrm{AND}, \epsilon, 1 \to 0]$ and satisfies $\max_{\mu\in\Omega_0} \mathrm{IC}_\mu(\pi) \le \mathrm{IC}^0(\mathrm{AND}, \epsilon, 1 \to 0) + \delta$ for some small $\delta > 0$. Consider the following protocol τ that computes DISJ_n with error.

- Alice and Bob exchange (with replacement using public randomness) $n^{2/3}$ random coordinates. Denote this set of random coordinates by J. If for some $j \in J$, $x_j = 1$ and $y_j = 1$, then they output 1 and terminate.
- For each coordinate outside J, Alice and Bob run the protocol π and output 1 if π outputs 1 on some coordinate. Otherwise they output 0.

As π has one-sided $1 \to 0$ error, obviously τ has only one-sided $1 \to 0$ error too, and this error happens with probability at most $\epsilon^d \le \epsilon$, where d is the number of coordinates outside J which satisfy $x_j = y_j = 1$ (if $x_j = y_j = 1$ for some coordinate in J, there is no error). In particular, τ computes $[\mathrm{DISJ}_n, \epsilon, 1 \to 0]$.

A direct inspection shows that the remaining proof of Lemma 8.5 in [4] depends only on the protocol but not on the specific problem, hence the proof works for our problem too, and the lemma can be proved similarly. ◀

Next we prove an amortized upper bound for DISJ_n.

▶ **Lemma 39.** *For every $\epsilon, \delta > 0$, there exists a constant $C > 0$ that depends on n, ϵ, δ, such that as long as $N \ge C(n, \epsilon, \delta)$, we have*

$$\frac{R_\epsilon(\mathrm{DISJ}_{n\times N})}{N} \le (1+\delta)\,\mathrm{IC}(\mathrm{DISJ}_n, \epsilon, 1 \to 0).$$

Proof. We sketch the proof below. More details can be found in Section 6.2 of [3].

- Step 1. Choose a good protocol for $[\mathrm{DISJ}_n, \epsilon - \xi, 1 \to 0]$ for an appropriate $\xi > 0$.
  Denote $I := \mathrm{IC}(\mathrm{DISJ}_n, \epsilon, 1 \to 0)$. By continuity of information complexity (Lemma 3, which holds for one-sided error with the same proof), there exists $\xi > 0$ such that

  $$\mathrm{IC}(\mathrm{DISJ}_n, \epsilon - \xi, 1 \to 0) \le \left(1 + \frac{\delta}{6}\right) I.$$

  A minimax argument along the lines of Theorem 3.5 and Theorem 3.6 of [3] (but simpler) shows that there exists a protocol π that computes $[\mathrm{DISJ}_n, \epsilon - \xi, 1 \to 0]$, and for every distribution μ, its information cost satisfies

  $$\mathrm{IC}_\mu(\pi) \le \left(1 + \frac{\delta}{3}\right) I.$$

  Denote by r the number of rounds in π.
- Step 2. Parallel computing.
  Let $M = \sqrt[3]{N}$. For an arbitrary distribution μ on $\{0,1\}^{n\times M} \times \{0,1\}^{n\times M}$, let $\mu_1, \ldots, \mu_M$ be the marginals of μ restricted to each block of size n. Consider π^M, that is, the execution of M copies of π in parallel. The protocol π^M has information cost

  $$\mathrm{IC}_\mu(\pi^M) \le \sum_{i=1}^{M} \mathrm{IC}_{\mu_i}(\pi) \le \left(1 + \frac{\delta}{3}\right) M \cdot I.$$

Clearly, π^M is still an r-round protocol (this is required in order to apply Braverman–Rao compression).

- Step 3. Compression (with the aid of a minimax argument), and truncation.
  By Braverman–Rao compression [6] one can find another protocol with communication cost roughly equal to $M \cdot I$, and with an extra small error. However, this extra error is distributional according to the distribution μ. What we want is to solve $[\mathrm{DISJ}_{n\times M}, \epsilon]$, that is, the protocol is only allowed to err with probability at most ϵ on *every* input. Fortunately, one can fix this by applying a minimax argument, presented as Claim 6.10 in [3], followed by an extra parallel computation step, presented as Claim 6.11 in [3].
  The analog of Claim 6.10 comes up with a protocol τ with the following properties:
  - For every input in $\{0,1\}^{n\times M} \times \{0,1\}^{n\times M}$, the statistical distance between the output of τ and the output of π^M is $O(1/M^3)$.
  - The expected communication cost of τ is at most $\left(1 + \frac{\delta}{2}\right) M \cdot I$.
  - The worst-case communication cost of τ is at most $O(Mn/\delta_1)$.

  (The statement of Claim 6.10 has $1/M^2$ instead of $1/M^3$, but the proof of Claim 6.10 works for any constant exponent; this can be traced to the fact that the dependence on the error in Braverman–Rao compression is logarithmic.)
  The idea now is to run M^2 copies of τ in parallel, truncating the result, as in Claim 6.11 of [3]. For large enough M (depending on n, ϵ, δ), the resulting protocol τ' satisfies the following properties:
  - For every input in $\{0,1\}^{n\times M\times M^2} \times \{0,1\}^{n\times M\times M^2}$, the statistical distance between the output of τ' and the output of τ^{M^2} is at most η, where η tends to zero as $M \to \infty$.
  - The worst-case communication complexity of τ' is at most $(1+\delta)M^3 \cdot I$.

  In particular, the statistical distance between τ' and $\pi^{M^3} = \pi^N$ is at most $\eta + O(1/M)$ on every input, which tends to zero as $M \to \infty$. Choose M large enough to guarantee that the statistical distance between the output of τ' and the output of π^N is at most ξ. The protocol τ' can be used to compute $[\mathrm{DISJ}_{n\times N}, \epsilon]$, as in the proof of Lemma 38. This completes the proof. ◀

Now we prove the upper bound.

Proof of Lemma 37. Fix $\epsilon > 0$. By Lemma 38, there exists $T(\epsilon)$ depending on ϵ such that

$$\mathrm{IC}(\mathrm{DISJ}_n, \epsilon, 1 \to 0) \leq n\,\mathrm{IC}^0(\mathrm{AND}, \epsilon, 1 \to 0) + o(n)$$

whenever $n \geq T(\epsilon)$. For every such sufficiently large n, choose $\delta = \frac{1}{n}$. Lemma 39 states that

$$\frac{R_\epsilon(\mathrm{DISJ}_{n\times N})}{N} \leq \left(1 + \frac{1}{n}\right) \mathrm{IC}(\mathrm{DISJ}_n, \epsilon, 1 \to 0)$$

whenever $N \geq C(n, \epsilon)$ for some constant $C(n, \epsilon)$. Since $\mathrm{IC}(\mathrm{DISJ}_n, \epsilon, 1 \to 0) \leq n$,

$$\frac{R_\epsilon(\mathrm{DISJ}_{n\times N})}{n \times N} \leq \mathrm{IC}^0(\mathrm{AND}, \epsilon, 1 \to 0) + \frac{1}{n} + o(1)$$

for $N \geq C(n, \epsilon)$. It follows that

$$\frac{R_\epsilon(\mathrm{DISJ}_M)}{M} \leq \mathrm{IC}^0(\mathrm{AND}, \epsilon, 1 \to 0) + o(1)$$

where $o(1) \to 0$ as $M \to \infty$, completing the proof. ◀

On input (X, Y):
- Alice and Bob, using public randomness, jointly sample a permutation σ on the set $\{1, 2, \ldots, n\}$ uniformly at random; and they run the following sub-protocol π^σ:
- For $i = 1, 2, \ldots, n$ repeat:
  - Alice and Bob run a protocol π_i^σ that is (almost) optimal for $\mathrm{IC}_{\nu_i}(\mathrm{AND}, \epsilon/2p, 1 \to 0)$ on input $(X_{\sigma(i)}, Y_{\sigma(i)})$, where ν_i is the distribution of $(X_{\sigma(i)}, Y_{\sigma(i)})$ conditioned on the event that the protocol has not yet terminated;
  - if the protocol π_i^σ outputs 1, then terminate and output 1;
- If the "for-loop" ends without outputting 1, output 0 and terminate.

Figure 2 The protocol π that solves $[\mathrm{DISJ}_n, \mu, \epsilon, 1 \to 0]$.

7.2 A protocol for Set-Disjointness

Theorem 15 (restated). *For the set-disjointness function* DISJ_n *on inputs of length* n*, we have*

$$\mathrm{IC}^D(\mathrm{DISJ}_n, \epsilon) = n[\mathrm{IC}^0(\mathrm{AND}, 0) - \Theta(\sqrt{h(\epsilon)})] + O(\log n).$$

Proof. We already established the lower bound in (14), it remains to prove the upper bound.

Let μ be an input distribution for DISJ_n, and let $p = \Pr_\mu[\mathrm{DISJ}_n(X, Y) = 1]$. We can assume that $p \geq \epsilon$ as otherwise $\mathrm{IC}_\mu(\mathrm{DISJ}_n, \mu, \epsilon) = 0$, and the upper bound trivially holds. Below we introduce a protocol π in Figure 2 that solves $[\mathrm{DISJ}_n, \mu, \epsilon]$ and has the desired information cost. In fact, our protocol is stronger in the sense that it has only one-sided error: the protocol π always outputs 0 correctly if the correct output is 0, and on the other hand, if there are $t \geq 1$ coordinates satisfying $X_i = Y_i = 1$, then π will erroneously output 0 with probability at most $(\epsilon/2p)^t \leq \epsilon/2p$. Thus the distributional error of π is at most $p \cdot \frac{\epsilon}{2p} < \epsilon$, and π indeed solves $[\mathrm{DISJ}_n, \mu, \epsilon, 1 \to 0]$.

We now analyze the information cost. We start by analyzing the information cost of the sub-protocol π^σ. Let Π^σ be the transcript of π^σ, and write $\Pi^\sigma = \Pi_1^\sigma \ldots \Pi_n^\sigma$ where Π_i^σ denotes the transcript of the protocol π_i^σ for $i = 1, \ldots, n$. As usual let $\Pi_{<i}^\sigma = \Pi_1^\sigma \ldots \Pi_{i-1}^\sigma$ be the partial transcript. Let μ_i denote the distribution of $X_{\sigma(i)}Y_{\sigma(i)}$, and ν_i denote the distribution of $X_{\sigma(i)}Y_{\sigma(i)}$ conditioned on $\Pi_{<i}^\sigma$. Corollary 11(iii) gives a bound on the information exchanged in each round: there exist constants $C_1, C_2 > 0$ such that for any distribution ν,

$$\mathrm{IC}_\nu(\mathrm{AND}, \epsilon/2p, 1 \to 0) \leq \mathrm{IC}^0(\mathrm{AND}, 0) + C_1 \overline{h}(\nu(1,1)) - C_2 \overline{h}(\epsilon/p).$$

Note that $(\Pi_i^\sigma | XY\Pi_{<i}^\sigma)$ has the same distribution as $(\Pi_i^\sigma | X_{\sigma(i)}Y_{\sigma(i)}\Pi_{<i}^\sigma)$, and thus

$$\begin{aligned} I(Y; \Pi^\sigma | X) &= \sum_{i=1}^n I(Y; \Pi_i^\sigma | X, \Pi_{<i}^\sigma) = \sum_{i=1}^n [H(\Pi_i^\sigma | X, \Pi_{<i}^\sigma) - H(\Pi_i^\sigma | XY, \Pi_{<i}^\sigma)] \\ &\leq \sum_{i=1}^n [H(\Pi_i^\sigma | X_{\sigma(i)}, \Pi_{<i}^\sigma) - H(\Pi_i^\sigma | X_{\sigma(i)}Y_{\sigma(i)}, \Pi_{<i}^\sigma)] \\ &= \sum_{i=1}^n I(Y_{\sigma(i)}; \Pi_i^\sigma | X_{\sigma(i)}, \Pi_{<i}^\sigma). \end{aligned}$$

Thus, denoting by T^σ the number of AND protocols executed before the termination of π^σ, the above inequality implies (note that ν_i is a random variable, and π_i^σ depends on ν_i)

$$\begin{aligned}\mathrm{IC}_\mu(\pi^\sigma) &\le \sum_{i=1}^n \mathbb{E}\,\mathrm{IC}_{\nu_i}(\pi_i^\sigma) \le \sum_{i=1}^n \Pr[T^\sigma \ge i]\,\mathbb{E}\left[\mathrm{IC}_{\nu_i}(\pi_i^\sigma) \mid T^\sigma \ge i\right] \\ &\le \sum_{i=1}^n \Pr[T^\sigma \ge i]\,\mathbb{E}\left[\mathrm{IC}^0(\mathrm{AND},0) + C_1\overline{h}(\nu_i(1,1)) - C_2\overline{h}(\epsilon/p) \mid T^\sigma \ge i\right] \\ &\le \left(\mathrm{IC}^0(\mathrm{AND},0) - C_2\overline{h}(\epsilon/p)\right)\mathbb{E}[T^\sigma] + C_1\sum_{i=1}^n \Pr[T^\sigma \ge i]\,\mathbb{E}\left[\overline{h}(\nu_i(1,1))|T^\sigma \ge i\right].\end{aligned}$$

We want to bound the second term. Note since $p \ge \epsilon$,

$$\begin{aligned}\Pr[T^\sigma = i | T^\sigma \ge i, X_{\sigma(i)} = Y_{\sigma(i)} = 1] &= \Pr[\pi_i^\sigma(X_{\sigma(i)}Y_{\sigma(i)}) = 1 | T^\sigma \ge i, X_{\sigma(i)} = Y_{\sigma(i)} = 1] \\ &\ge 1 - \frac{\epsilon}{2p} \\ &\ge 1/2.\end{aligned}$$

Hence, applying (12) twice and using the concavity of $\overline{h}$, we get

$$\begin{aligned}&\Pr[T^\sigma \ge i]\,\mathbb{E}\left[\overline{h}(\nu_i(1,1))|T^\sigma \ge i\right] \\ \le\ &\Pr[T^\sigma \ge i]\overline{h}\left(\mathbb{E}\left[\nu_i(1,1)|T^\sigma \ge i\right]\right) \\ =\ &\Pr[T^\sigma \ge i]\overline{h}(\Pr[X_{\sigma(i)} = Y_{\sigma(i)} = 1 | T^\sigma \ge i]) \\ \le\ &\overline{h}(\Pr[X_{\sigma(i)} = Y_{\sigma(i)} = 1 | T^\sigma \ge i]\Pr[T^\sigma \ge i]) \\ =\ &\overline{h}(\Pr[T^\sigma \ge i, X_{\sigma(i)} = Y_{\sigma(i)} = 1]) \\ \le\ &2\Pr[T^\sigma = i | T^\sigma \ge i, X_{\sigma(i)} = Y_{\sigma(i)} = 1]\overline{h}(\Pr[T^\sigma \ge i, X_{\sigma(i)} = Y_{\sigma(i)} = 1]) \\ \le\ &2\overline{h}(\Pr[T^\sigma = i, X_{\sigma(i)} = Y_{\sigma(i)} = 1]) \\ \le\ &2\overline{h}(\Pr[T^\sigma = i, \pi(X,Y) = 1]).\end{aligned}$$

Using concavity of $\overline{h}$ again,

$$\frac{1}{n}\sum_{i=1}^n \overline{h}(\Pr[T^\sigma = i, \pi(X,Y) = 1]) \le \overline{h}(\Pr[\pi(X,Y) = 1]/n) = \overline{h}(p/n).$$

Therefore

$$\sum_{i=1}^n \Pr[T^\sigma \ge i]\,\mathbb{E}\left[\overline{h}(\nu_i(1,1))|T^\sigma \ge i\right] \le 2n\overline{h}(p/n).$$

That is, we have shown

$$\mathrm{IC}_\mu(\pi^\sigma) \le \left(\mathrm{IC}^0(\mathrm{AND},0) - C_2\overline{h}(\epsilon/p)\right)\mathbb{E}[T^\sigma] + 2C_1 n\overline{h}(p/n). \tag{54}$$

Taking the expectation with respect to σ, we obtain

$$\mathrm{IC}_\mu(\pi) = \mathop{\mathbb{E}}_\sigma \mathrm{IC}_\mu(\pi^\sigma) = \left(\mathrm{IC}^0(\mathrm{AND},0) - C_2\overline{h}(\epsilon/p)\right)\mathop{\mathbb{E}}_{\sigma,XY}[T^\sigma] + 2C_1 n\overline{h}(p/n). \tag{55}$$

Hence it remains to bound $\mathbb{E}[T^\sigma]$ where the expectation is over σ and the input XY.

Let x, y be such that $\mathrm{DISJ}(x,y) = 1$, and let j be an index such that $\mathrm{AND}(x_j, y_j) = 1$. Then

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\sigma,XY}[T^\sigma | XY = xy] \\
&= \sum_{i=1}^{n} \Pr[\sigma(i) = j]\, \mathbb{E}[T^\sigma | XY = xy, \sigma(i) = j] \\
&= \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}[T^\sigma | XY = xy, \sigma(i) = j] \\
&= \frac{1}{n} \sum_{i=1}^{n} \sum_{b=0,1} \mathbb{E}[T^\sigma | XY = xy, \sigma(i) = j, \pi_i^\sigma(X,Y) = b] \Pr[\pi_i^\sigma(X,Y) = b | XY = xy, \sigma(i) = j] \\
&\le \frac{1}{n} \sum_{i=1}^{n} \Big(i \Pr[\pi_i^\sigma(X,Y) = 1 | XY = xy, \sigma(i) = j] + n \Pr[\pi_i^\sigma(X,Y) = 0 | XY = xy, \sigma(i) = j] \Big) \\
&\le \frac{1}{n} \sum_{i=1}^{n} \left(i(1 - \frac{\epsilon}{2p}) + n\frac{\epsilon}{2p} \right) = (1 - \frac{\epsilon}{2p})\frac{n+1}{2} + \frac{\epsilon}{2p} n \le \frac{n+1}{2} + \frac{\epsilon}{4p} n.
\end{aligned}$$

This allows us the next bound:

$$\begin{aligned}
\mathop{\mathbb{E}}_{\sigma,XY}[T^\sigma] &= \Pr[\mathrm{DISJ}(X,Y) = 1]\, \mathbb{E}[T | \,\mathrm{DISJ}(X,Y) = 1] \\
&\qquad + \Pr[\mathrm{DISJ}(X,Y) = 0]\, \mathbb{E}[T | \,\mathrm{DISJ}(X,Y) = 0] \\
&\le p \left(\frac{n+1}{2} + \frac{\epsilon}{4p} n \right) + (1-p)n \le \frac{2p}{3} n + \frac{\epsilon}{4} n + (1-p)n \\
&= (1 - p/3 + \epsilon/4)n. \qquad (56)
\end{aligned}$$

Combine (55) and (56) we get

$$\begin{aligned}
\mathrm{IC}_\mu(\pi) &\le n(1 - p/3 + \epsilon/4) \left(\mathrm{IC}^0(\mathrm{AND}, 0) - C_2 \overline{h}(\epsilon/p) \right) + C_1 2n\overline{h}(p/n) \\
&= n(\mathrm{IC}_0(\mathrm{AND}, 0) - \Omega(\overline{h}(\epsilon/p) + p)) + O(n\overline{h}(p/n)).
\end{aligned}$$

It remains to optimize over p. We start by minimizing $p + \overline{h}(\epsilon/p)$. Up to a constant multiple, the minimum is attained at the point satisfying $p = \overline{h}(\epsilon/p)$. A simple calculation shows that $p \approx \sqrt{h(\epsilon)}$, and so $p + \overline{h}(\epsilon/p) = \Omega(\sqrt{h(\epsilon)})$. Thus

$$\mathrm{IC}_\mu(\pi) \le n[\mathrm{IC}^0(\mathrm{AND}, 0) - \Omega(\sqrt{h(\epsilon)})] + O(n\overline{h}(p/n)).$$

The value of the error term $O(n\overline{h}(p/n))$ is at most $O(n\overline{h}(1/n)) = O(n\frac{\log n}{n}) = O(\log n)$, and the theorem follows. ◀

8 Open problems and concluding remarks

- In Conjecture 14 we speculated that the exact asymptotics of $R_\epsilon(\mathrm{DISJ}_n)$ is given by the information complexity of the AND function when only one-sided error is allowed:

  $$R_\epsilon(\mathrm{DISJ}_n) = n\, \mathrm{IC}^0(\mathrm{AND}, \epsilon, 1 \to 0) \pm o(n).$$

  The set disjointness function has a "self-reducible" structure in the sense that it is possible to solve an instance of the corresponding communication problem by dividing the input into blocks and solving the same problem on each block separately. This structure

allows us to relate the communication complexity of the problem to its amortized communication complexity, and thus to its information complexity via the fundamental result of Braverman and Rao [6]. Applying such ideas we showed (the lower bound is obvious)

$$\mathrm{IC}(\mathrm{DISJ}_n, \epsilon) \leq R_\epsilon(\mathrm{DISJ}_n) \leq m\,\mathrm{IC}(\mathrm{DISJ}_{\frac{n}{m}}, \epsilon, 1 \to 0) + o(n),$$

for an appropriate choice of $m = m(n)$ that tends to infinity as $n \to \infty$. In Theorem 13 we combined this with our analysis of the information complexity of the set disjointness to prove $R_\epsilon(\mathrm{DISJ}_n) = n[\mathrm{IC}^0(\mathrm{AND}, 0) - \Theta(h(\epsilon))]$. More precisely we showed

$$n\,\mathrm{IC}^0(\mathrm{AND}, \epsilon) \leq \mathrm{IC}(\mathrm{DISJ}_n, \epsilon) \leq \mathrm{IC}(\mathrm{DISJ}_n, \epsilon, 1 \to 0) \leq n\,\mathrm{IC}^0(\mathrm{AND}, \epsilon, 1 \to 0) + o(n),$$

and combined it with our results regarding the information complexity of the AND function. We believe that the upper bound is the truth; that is

$$\mathrm{IC}(\mathrm{DISJ}_n, \epsilon) \geq n\,\mathrm{IC}^0(\mathrm{AND}, \epsilon, 1 \to 0) - o(n),$$

which would imply Conjecture 14.

- The example of the AND function shows that the $\Omega(h(\epsilon))$ gain in the information cost, appearing in our upper bounds in Theorems 5, 8, 17 and 18 is tight. However we do not know whether the $O(h(\sqrt{\epsilon}))$ gain appearing in the lower bounds in Theorems 7 and 8, Corollary 16 and Theorem 18 is sharp. In fact we are not aware of any example that exhibits a gain that is not $\Theta(h(\epsilon))$. Is it true that for every function $f\colon \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$, and measure μ on $\mathcal{X} \times \mathcal{Y}$ with $\mathrm{IC}_\mu(f, 0) > 0$, we have $\mathrm{IC}_\mu(f, \epsilon) = \mathrm{IC}_\mu(f, 0) - \Theta(h(\epsilon))$? One can ask a similar question for $\mathrm{IC}_\mu(f, \mu, \epsilon)$, $\mathrm{IC}(f, \epsilon)$, and $\mathrm{IC}^D(f, \epsilon)$.
- Recall that the *inner product function* $\mathrm{IP}_n\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ is defined as

  $$\mathrm{IP}_n\colon (x, y) \mapsto \sum_{i=1}^{n} x_i y_i \mod 2.$$

  Let ν denote the uniform probability measure on $\{0,1\}^n \times \{0,1\}^n$. It is easy to see that $\mathrm{IC}_\nu(\mathrm{IP}_n, \nu, \epsilon) \leq (1 - 2\epsilon)n$. In [5, Theorem 1.3], Braverman et al. exploited the self-reducibility properties of the inner product function to showed that for every $\delta > 0$, there exists an $\epsilon > 0$ and $n_0 > 0$ such that for every $n > n_0$, $\mathrm{IC}(\mathrm{IP}_n, \epsilon) > (1 - \delta)n$.
  In [5, Problem 1.4] they ask whether the dependency of δ on ϵ is linear. In other words, is there a constant $\alpha > 0$ such that for every sufficiently small $\epsilon > 0$ and sufficiently large n, $\mathrm{IC}_\nu(\mathrm{IP}_n, \nu, \epsilon) \geq (1 - \alpha\epsilon)n$? If yes, then can we take $\alpha \approx 2$, or more precisely, is it true that $\mathrm{IC}_\nu(\mathrm{IP}_n, \nu, \epsilon) = (1 - 2\epsilon - o(\epsilon))n$? Note that the bound $\mathrm{IC}_\nu(f, \nu, \epsilon) < \mathrm{IC}_\nu(f, \nu, 0) - \Omega(h(\epsilon))$ of Theorem 8 does not refute these possibilities as in these questions ϵ is fixed, and asymptotics are as $n \to \infty$.
- The focus of this paper has been on the internal information complexity, and except for few results such as Proposition 6, we have not studied the external information complexity analogues. However considering that external information complexity is typically simpler than internal information complexity, we believe that the analogues of many of our results, specially those about the AND function, can be proven for this case as well. We defer this to future research.

References

1 Ziv Bar-Yossef, T. S. Jayram, Ravi Kumar, and D. Sivakumar. An information statistics approach to data stream and communication complexity. *J. Comput. System Sci.*, 68(4):702–732, 2004. doi:10.1016/j.jcss.2003.11.006.

2 Boaz Barak, Mark Braverman, Xi Chen, and Anup Rao. How to compress interactive communication [extended abstract]. In *STOC'10 – Proceedings of the 2010 ACM International Symposium on Theory of Computing*, pages 67–76. ACM, New York, 2010.

3 Mark Braverman. Interactive information complexity. In *STOC'12 – Proceedings of the 2012 ACM Symposium on Theory of Computing*, pages 505–524. ACM, New York, 2012. doi:10.1145/2213977.2214025.

4 Mark Braverman, Ankit Garg, Denis Pankratov, and Omri Weinstein. From information to exact communication (extended abstract). In *STOC'13 – Proceedings of the 2013 ACM Symposium on Theory of Computing*, pages 151–160. ACM, New York, 2013. doi:10.1145/2488608.2488628.

5 Mark Braverman, Ankit Garg, Denis Pankratov, and Omri Weinstein. Information lower bounds via self-reducibility. In Andrei A. Bulatov and Arseny M. Shur, editors, *Computer Science – Theory and Applications*, volume 7913 of *Lecture Notes in Computer Science*, pages 183–194. Springer Berlin Heidelberg, 2013. doi:10.1007/978-3-642-38536-0_16.

6 Mark Braverman and Anup Rao. Information equals amortized communication. *IEEE Trans. Inform. Theory*, 60(10):6058–6069, 2014. doi:10.1109/TIT.2014.2347282.

7 Mark Braverman, Anup Rao, Omri Weinstein, and Amir Yehudayoff. Direct product via round-preserving compression. In *Automata, languages, and programming. Part I*, volume 7965 of *Lecture Notes in Comput. Sci.*, pages 232–243. Springer, Heidelberg, 2013. doi:10.1007/978-3-642-39206-1_20.

8 Mark Braverman, Anup Rao, Omri Weinstein, and Amir Yehudayoff. Direct products in communication complexity. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science – FOCS 2013*, pages 746–755. IEEE Computer Soc., Los Alamitos, CA, 2013. doi:10.1109/FOCS.2013.85.

9 Mark Braverman and Jon Schneider. Information complexity is computable. *CoRR*, abs/1502.02971, 2015. URL: http://arxiv.org/abs/1502.02971.

10 Amit Chakrabarti, Yaoyun Shi, Anthony Wirth, and Andrew Yao. Informational complexity and the direct sum problem for simultaneous message complexity. In *42nd IEEE Symposium on Foundations of Computer Science (Las Vegas, NV, 2001)*, pages 270–278. IEEE Computer Soc., Los Alamitos, CA, 2001.

11 Arkadev Chattopadhyay and Toniann Pitassi. The story of set disjointness. *ACM SIGACT News*, 41(3):59–85, 2010.

12 Yuval Dagan and Yuval Filmus. Grid protocols. In preparation, 2016.

13 Tomás Feder, Eyal Kushilevitz, Moni Naor, and Noam Nisan. Amortized communication complexity. *SIAM J. Comput.*, 24(4):736–750, 1995. doi:10.1137/S0097539792235864.

14 Yuval Filmus, Hamed Hatami, Yaqiao Li, and Suzin You. Information complexity of the and function in the two-party, and multiparty settings. Submitted, 2017.

15 Anat Ganor, Gillat Kol, and Ran Raz. Exponential separation of information and communication for Boolean functions [extended abstract]. In *STOC'15 – Proceedings of the 2015 ACM Symposium on Theory of Computing*, pages 557–566. ACM, New York, 2015.

16 Prahladh Harsha, Rahul Jain, David McAllester, and Jaikumar Radhakrishnan. The communication complexity of correlation. *IEEE Trans. Inform. Theory*, 56(1):438–449, 2010. doi:10.1109/TIT.2009.2034824.

17 Rahul Jain. New strong direct product results in communication complexity. *J. ACM*, 62(3):Art. 20, 27, 2015. doi:10.1145/2699432.

18 Rahul Jain, Attila Pereszlényi, and Penghui Yao. A direct product theorem for the two-party bounded-round public-coin communication complexity. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science – FOCS 2012*, pages 167–176. IEEE Computer Soc., Los Alamitos, CA, 2012.

19 Rahul Jain, Jaikumar Radhakrishnan, and Pranab Sen. A direct sum theorem in communication complexity via message compression. In *Automata, languages and programming*, volume 2719 of *Lecture Notes in Comput. Sci.*, pages 300–315. Springer, Berlin, 2003. doi:10.1007/3-540-45061-0_26.

20 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.

21 Hartmut Klauck. A strong direct product theorem for disjointness [extended abstract]. In *STOC'10 – Proceedings of the 2010 ACM International Symposium on Theory of Computing*, pages 77–86. ACM, New York, 2010.

22 Eyal Kushilevitz and Noam Nisan. *Communication complexity*. Cambridge University Press, Cambridge, 1997.

23 Nan Ma and Prakash Ishwar. Some results on distributed source coding for interactive function computation. *IEEE Trans. Inform. Theory*, 57(9):6180–6195, 2011. doi:10.1109/TIT.2011.2161916.

24 Nan Ma and Prakash Ishwar. The infinite-message limit of two-terminal interactive source coding. *IEEE Trans. Inform. Theory*, 59(7):4071–4094, 2013. doi:10.1109/TIT.2013.2251412.

25 Marco Molinaro, David P. Woodruff, and Grigory Yaroslavtsev. Beating the direct sum theorem in communication complexity with implications for sketching. In *Proceedings of the Twenty-fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'13, pages 1738–1756, Philadelphia, PA, USA, 2013. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=2627817.2627942.

26 A. A. Razborov. On the distributional complexity of disjointness. *Theoret. Comput. Sci.*, 106(2):385–390, 1992. doi:10.1016/0304-3975(92)90260-M.

27 Byron Schmuland. On the compacity of the space of probability measures. Mathematics Stack Exchange. URL: https://math.stackexchange.com/q/642888.

28 C. E. Shannon. A mathematical theory of communication. *Bell System Tech. J.*, 27:379–423, 623–656, 1948.

29 Alexander A. Sherstov. Communication complexity theory: thirty-five years of set disjointness. In *Mathematical foundations of computer science 2014. Part I*, volume 8634 of *Lecture Notes in Comput. Sci.*, pages 24–43. Springer, Heidelberg, 2014. doi:10.1007/978-3-662-44522-8_3.

30 Andrew Chi-Chih Yao. Some complexity questions related to distributive computing(preliminary report). In *Proceedings of the Eleventh Annual ACM Symposium on Theory of Computing*, STOC'79, pages 209–213, New York, NY, USA, 1979. ACM. doi:10.1145/800135.804414.

Stochasticity in Algorithmic Statistics for Polynomial Time*

Alexey Milovanov[1] and Nikolay Vereshchagin[2]

1 **National Research University Higher School of Economics, Moscow, Russia**
almas239@gmail.com
2 **National Research University Higher School of Economics, Moscow, Russia**
nikolay.vereshchagin@gmail.com

Abstract

A fundamental notion in Algorithmic Statistics is that of a stochastic object, i.e., an object having a simple plausible explanation. Informally, a probability distribution is a plausible explanation for x if it looks likely that x was drawn at random with respect to that distribution. In this paper, we suggest three definitions of a plausible statistical hypothesis for Algorithmic Statistics with polynomial time bounds, which are called *acceptability, plausibility and optimality*. Roughly speaking, a probability distribution μ is called an acceptable explanation for x, if x possesses all properties decidable by short programs in a short time and shared by almost all objects (with respect to μ). Plausibility is a similar notion, however this time we require x to possess all properties T decidable even by long programs in a short time and shared by almost all objects. To compensate the increase in program length, we strengthen the notion of 'almost all' – the longer the program recognizing the property is, the more objects must share the property. Finally, a probability distribution μ is called an optimal explanation for x if $\mu(x)$ is large (close to $2^{-\mathrm{C}^{\mathrm{poly}}(x)}$).

Almost all our results hold under some plausible complexity theoretic assumptions. Our main result states that for acceptability and plausibility there are infinitely many non-stochastic objects, i.e. objects that do not have simple plausible (acceptable) explanations. Using the same techniques, we show that the distinguishing complexity of a string x can be super-logarithmically less than the conditional complexity of x with condition r for almost all r (for polynomial time bounded programs). Finally, we study relationships between the introduced notions.

1998 ACM Subject Classification F.2.0 Analysis of Algorithms and Problem Complexity

Keywords and phrases Algorithmic Statistics, Kolmogorov complexity, elusive set, distinguishing complexity

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.17

1 Introduction

Acceptable statistical hypotheses

▶ **Example 1.** Assume we are given an n-bit natural number x which is a square and has no other features. Which statistical hypotheses we would accept for x? An acceptable hypothesis is the following: the number x was obtained by random sampling in the set of all n-bit squares, where all numbers have equal chances to be drawn (the hypothesis μ_1).

* The first author is supported in part by Young Russian Mathematics award. The work of both authors was in part supported by the RFBR grant 16-01-00362, by the Russian Academic Excellence Project '5-100' and by an ANR grant RaCAF ANR-15-CE40-0016-01 .

And the following hypothesis μ_2 is clearly not acceptable: the number x was obtained by random sampling in the set of *all* n-bit numbers. On what grounds we refute hypothesis μ_2? Because we can exhibit an easily checked property (to be a square) possessed by x and not possessed by a vast majority of all n-bit strings.

The reader can object to this line of reasoning by noting that on these grounds we can reject the hypothesis μ_1 as well. Indeed, we exhibit the property "to be equal to x", which is also shared by a negligible fraction of numbers with respect to μ_1. However, in contrast to the property "to be a square", this property is not simple, as it has no short program recognizing the property in a short time. And for the property "to be a square", there is such a program.

Generalizing this example, we will define the notion of an acceptable statistical hypothesis x. A probability distribution μ over the set of binary strings will be called an *acceptable hypothesis* for a binary string x if there is no simple set $T \ni x$ with negligible $\mu(T)$. We will call a set T *simple* if there is a short program to decide membership in T in a short time, as in Example 1.

A string will be called *stochastic*, if it has a simple acceptable hypothesis. How will we measure simplicity of a probability distribution μ? In the same way as we measure the simplicity of a refutation set T: a probability distribution will be called simple, if it can be sampled by a short probabilistic machine with no input in a short time. We say that such a machine *samples a distribution* μ, if for all x the probability of the event "M outputs x" equals $\mu(x)$. The running time of M is defined as the maximum of M's running time over all outcomes of its coin tossing.

Of course in a rigorous definition of an acceptable hypothesis μ, we have to specify three parameters: the upper bound α for the length of a program that recognizes T, the upper bound t for the running time of that program, and the upper bound ε for $\mu(T)$ (how small should be $\mu(T)$ to be qualified as "negligible"). The larger these parameters are, the stronger the notion of an acceptable hypothesis is. And in a rigorous definition of a simple distribution μ, we have to specify two parameters: the upper bound α' for the length of a program sampling μ and the upper bound t' for the running time of that program. The smaller these parameters are, the stronger the notion of a simple distribution is. Thus in the notion of stochasticity we have 5 parameters, α', t' and α, t, ε. It seems natural to choose $\alpha > \alpha'$ and $t > t'$, that is, to give more resources to those who want to refute a hypothesis μ than the amount of resources needed to sample μ (as it was in Example 1).

Also in the definition of an acceptable hypothesis the parameter ε should be much smaller than $2^{-\alpha}$. In this case the notion of an acceptable distribution satisfies *The Majority Principle: for every probability distribution μ for almost all (w.r.t. μ) strings x the distribution μ is an acceptable hypothesis for x* (Proposition 7 below). We believe that any notion of a plausible statistical hypothesis should satisfy this principle.

The main question we are interested in is the following: for which values of parameters there are strings that have no simple acceptable explanations? Such strings will be called *non-stochastic*. Our main result states that under assumption NE $\neq$ RE *there are infinitely many non-stochastic strings x for $t, t', 1/\varepsilon = poly(n)$ and $\alpha, \alpha' = O(\log n)$[1], where $n = |x|$* (Theorem 8).

In Section 6 we explain why we need complexity theoretic assumptions to prove the main result: we prove that existence of non-stochastic strings for such parameters implies that

[1] All of the logarithms are base 2.

P $\neq$ PSPACE. To prove Theorem 8, we introduce the notion of an *elusive set*. Using that notion, we establish that there is a super-logarithmic gap between Kolmogorov complexity and Distinguishing complexity with polynomial time bounds (similar questions were addressed in [3]). We also study the following two notions of a good statistical hypothesis.

Plausible statistical hypotheses

▶ **Example 2.** Let $G : \{0,1\}^n \to \{0,1\}^{2n}$ be a Pseudo Random Number Generator (the precise definition is given in Assumption 4 below). Consider a string $x = G(s)$ of length $2n$. Would we accept the uniform distribution over all strings of length $2n$ as a good statistical hypothesis for x? We do not like this hypothesis, as the fraction of x' of length $2n$ for which $x' = G(s')$ for some s' is negligible. However it is impossible to check this property by a short program in short time – for almost all s the uniform distribution over all strings of length $2n$ is an acceptable hypothesis for $G_n(s)$ (Theorem 17). However for every fixed s the property $G_n(s) = x$ can be decided by a *long* program (of length n), into which s is hard-wired.

Let us give up the requirement that the program recognizing T in a short time is short. In a compensation, let us decrease the threshold for $\mu(T)$: we will now think that $\mu(T)$ is negligible if $\log \mu(T)$ is much less than the negative length of the program recognizing T. Notice that in Example 2 we have $\log \mu(T) = -2n$, which is by n less than the negative length of the program recognizing T. Probability distributions satisfying this requirement are called *plausible hypotheses for* x. The definitions imply that every plausible hypothesis is acceptable (Proposition 6). The converse is false (Theorem 17). And again the notion of plausibility satisfies the Majority Principle (Proposition 7).

As plausibility implies acceptability, our main result implies that there are infinitely many strings that have no simple plausible explanations. The existence of such strings can be proved also under other assumptions. Indeed, under Assumption 2 (below) only strings whose distinguishing complexity is close to Kolmogorov complexity can have simple plausible explanations (Proposition 16). And strings with a large gap between these complexities exist under assumption FewP $\cap$ SPARSE $\not\subseteq$ P [3].

Optimal statistical hypotheses

In practice, it is hard to decide whether a given probability distribution μ is plausible or acceptable for a given string x, as there are many possible "refutation sets" T and for a given T it is very hard to check whether it indeed refutes μ or not. Ideally, we would like to have a sound notion of a good hypothesis such that for a given simple distribution μ and a given string x, we could decide whether μ is good for x in a short time. Or, at least to refute μ in a short time, if μ is not good for x.

There is a natural parameter measuring how good is μ as an explanation for x, namely $\mu(x)$. Let us try to use this parameter instead of "refutation sets". According to the new definition, a simple probability distribution μ is a good explanation for x if $\mu(x)$ is large. How large? We will compare $\mu(x)$ with $2^{-\mathrm{C}^t(x)}$, where $\mathrm{C}^t(x)$ denotes Kolmogorov complexity with time bounded by t, where t is large enough compared to the running time of the short probabilistic program sampling μ. We will call μ an *optimal hypothesis for* x if $\mu(x) \approx 2^{-\mathrm{C}^t(x)}$.

There are three arguments to justify this definition. Firstly, whatever t we choose, the Majority Principle holds true (Proposition 13). Second, under some complexity theoretic assumption, if t is large enough compared to the running time of probabilistic machine sampling μ then $\mu(x)$ cannot significantly exceed $2^{-\mathrm{C}^t(x)}$, therefore, if $\mu(x)$ is close to this

value, then μ is optimal indeed. This fact was established in [1]. And third, given μ, x we can prove in a short time that μ is not an optimal hypothesis for x, if this is the case – it suffices to produce a program p for x such that $\mu(x) \ll 2^{-|p|}$ (we assume here that $\mu(x)$ can be computed in a short time).

Relations between the introduced notions

It follows from the definitions that plausibility implies acceptability and optimality. (To prove the second implication, we let $T = \{x\}$ in the definition of plausibility.) All other statements in the remainder of this section hold true under some assumptions (that are specified later).

For strings x with $\mathrm{CD}^{\mathrm{poly}(n)}(x) \ll \mathrm{C}^{\mathrm{poly}(n)}(x)$ there are no plausible explanations at all (Proposition 16). For such strings we are not aware about any relations between acceptability and optimality.

On the other hand, for strings x with $\mathrm{CD}^{\mathrm{poly}(n)}(x) \approx \mathrm{C}^{\mathrm{poly}(n)}(x)$, the picture is clear: Plausibility = Optimality $\Rightarrow$ Acceptability, and the converse implication does not hold (Example 2, Theorem 17 and Remark 3.3). The equivalence of plausibility and optimality (Theorem 18) for such strings is good news, as it provides a justification to the Maximal Likelihood Estimator. Indeed, imagine that x was drawn at random w.r.t. a simple but unknown probability distribution μ. Then with high μ-probability all $\mathrm{C}(x), \mathrm{C}^t(x), \mathrm{CD}^t(x)$ are close to each other and are close to $-\log \mu(x)$ [2] and μ is an acceptable and plausible hypothesis for x (Propositions 7, 13, 12). Given x, we want to find μ or any other plausible or at least acceptable statistical hypothesis for x. Using the Maximal Likelihood Estimator, we choose among all simple hypotheses μ the one that maximizes $\mu(x)$. Theorem 18 guarantees the success to this strategy – the chosen hypothesis μ is both acceptable and plausible.

In the next section we define the notions of a program and of Kolmogorov complexities used in the paper.

2 Preliminaries

Fix a deterministic one-tape Turing machine U that inputs three binary strings p, x, y and outputs one binary string and satisfies the following condition:

> For any other deterministic one-tape Turing machine V there is a constant c and a polynomial f such that for all p there is q with $|q| < |p| + c$ for which $U(q, y, r) = V(p, y, r)$ (for all y, r) and the running time of $U(q, y, r)$ is bounded by
>
> $$f(|y| + |r| + \text{ the running time of } V(p, y, r))$$
>
> and the similar inequality for the space holds as well.

This machine will be called *universal*. Using the universal machine we can define the Kolmogorov complexity (with or without time or space bounds) and the notions of programs and their running times for deterministic and randomized machines.

Kolmogorov complexity: $\mathrm{C}^t(x|y)$ is the minimal length of p such that $U(p, y, \Lambda) = x$ in time t. Similarly, $\mathrm{CS}^m(x|y)$ is the minimal length of p such that $U(p, y, \Lambda) = x$ on space s. If $U(p, y, \Lambda) = x$ in time t, we say that p is a *program for x conditional to y* (or simply a *program for x*, if $y = \Lambda$), and we call t *the running time of p* on input y.

[2] Provided that t is larger than certain polynomial of the time needed to sample μ.

The distinguishing complexity: $\mathrm{CD}^t(x|y)$ is the minimal length of p such that
- $U(p,x,y)=1$;
- $U(p,x',y)$ halts in t steps for all x' of the same length as x;
- $U(p,x',y)=0$ for all $x'\neq x$.

Programs of deterministic machines: We say that a *program p outputs y on input x in time t* if $U(p,y,\Lambda)$ in time t.

Programs of randomized machines: Considering the uniform probability distribution over r's, we obtain a universal randomized machine. More specifically, a *program of a randomized machine* is a pair (p,m). A machine with program (p,m) on an input string y tosses a fair coin m times and then outputs $U(p,y,r)$, where r denotes the outcome of the tossing. We can fix $y=\Lambda$ thus obtaining the notion of a program of a randomized machine without input. The length of the program (p,m) is defined as $|p|+\log m$, and the running time (space) as the maximum over all $r\in\{0,1\}^m$ of the running time (space) of $U(p,y,r)$.

In the next section provide the rigorous definitions and formulations to all informal definitions and statements mentioned in the Introduction.

3 Our results and their comparison to the previous ones

3.1 Existence of non-stochastic strings

By a technical reason we consider only probability distributions μ over $\{0,1\}^n$ for some n and assume that $\mu(x)$ is a rational number for all x.

▶ **Definition 3.** Let t,α be natural numbers and ε a real number between 0 and 1. A (t,α,ε)-*acceptable* statistical hypothesis (or *explanation*) for a string x of length n is a probability distribution μ such that $\mu(T)\geqslant\varepsilon$ for all $T\ni x$ recognized by a deterministic program of length less than α in at most t steps for all inputs of length n.

The larger t,α,ε are, the stronger the notion of a (t,α,ε)-acceptable hypothesis becomes. For every x the distribution concentrated on x is a $(*,*,1)$-acceptable hypothesis for x (the asterisk for the time parameter means that the time can be arbitrary large as long as the program always halts). However, we are interested in simple explanations.

▶ **Definition 4.** A probability distribution μ is called (t,α)-*simple* if it can be sampled by a probabilistic program (with no input) of length less than α in time at most t.[3] (Recall that a machine M samples μ in time t if for all x the probability of event "M outputs x" equals $\mu(x)$ and the running time of M is at most t for all outcomes of coin tossing.)

Strings that have (t',α')-simple (t,α,ε)-acceptable explanations for small t',α' and large t,α,ε are informally called *stochastic* and otherwise *non-stochastic*. The smaller t',α' and the larger t,α,ε are, the stronger the notion of stochasticity is and the weaker the notion of non-stochasticity is.

▶ **Definition 5.** A probability distribution μ is called a (t,ε)-*plausible* hypothesis for a sting x of length n, if for any set $T\ni x$ recognized by a program of length l whose running time on all inputs of length n is at most t we have $\mu(T)\geqslant 2^{-l}\varepsilon$.

[3] In Theorem 18 we will need simplicity in another sense: we will need that the function $x\mapsto\mu(x)$ can be computed by a program of length α in time t.

The following proposition is a straightforward corollary from the definitions:

▶ **Proposition 6.** *Every (t,ε)-plausible hypothesis for x is $(t,\alpha,\varepsilon 2^{-\alpha})$-acceptable for x for any α.*

▶ Remark. Notice that if μ is (t,α)-plausible for x then $\mu(x) > 0$. In contrast, (t,α,ε)-acceptability of μ for x does not imply that in general. However, if the set $T = \{x \mid \mu(x) = 0\}$ can be recognized by a program of length α in time t, then (t,α,ε)-acceptability for x implies $\mu(x) > 0$. Another reason for not stipulating $\mu(x) > 0$ in the definition of acceptability is that we can achieve this almost 'for free'. Indeed, for every (t,α)-simple distribution μ the distribution μ' that is the arithmetic mean of μ and the uniform distribution over the set of all strings of length n is (t',α')-simple for t',α' close to t,α. For all x of length n we have $\mu'(x) > 0$. If μ is (t,α,ε)-acceptable for x, then μ' is $(t,\alpha,\varepsilon/2)$-acceptable for x.

The next statement shows that the Majority Principle is valid for (t,α,ε)-acceptability provided $\varepsilon \ll 2^{-\alpha}$ and for (t,ε)-plausibility provided $\varepsilon \ll 1/n$.

▶ **Proposition 7** (Majority Principle)**.** *For every probability distribution μ over binary strings of length n and all α,ε we have*

$$\mu\{x \mid \mu \text{ is not } (*,\alpha,\varepsilon)\text{-acceptable for } x\} < \varepsilon 2^{\alpha},$$
$$\mu\{x \mid \mu \text{ is not } (*,\varepsilon)\text{-plausible for } x\} < \varepsilon(n + O(1)).$$

This proposition as well as all other statements in this section will be proved in Section 5.

Our main result shows that there are infinitely many *non*-stochastic strings x for polynomial values of $t,t',1/\varepsilon$ and logarithmic values of α,α'. This result holds under the following

▶ Assumption 1. For some language L in NP over the unary alphabet there is no probabilistic polynomial time machine that for each string x in L finds a certificate for membership of x in L with probability at least $1/2$. Equivalently, for some language L in NE (the class of languages accepted in time $2^{O(n)}$ by non-deterministic Turing machines) there is no probabilistic machine that for each string x in L in time $2^{O(|x|)}$ finds a certificate for membership of x in L with probability at least $1/2$.

This assumption follows from the assumption NE $\neq$ RE, where RE denotes the class of languages recognized in time $2^{O(n)}$ by probabilistic Turing machines that err with probability at most $1/2$ for all strings in the language and do not err for strings outside the language. It is unknown whether these two assumptions are equivalent or not (see [5]).

▶ **Theorem 8.** *Under Assumption 1 for some constant d for all c for infinitely many n there is a string of length n that has no $(n^c, c\log n)$-simple (n^d, d, n^{-c})-acceptable hypotheses.*

In other words, for the strings x from this theorem, for every $(n^c, c\log n)$-simple μ there is $T \ni x$ recognized by a program of length d in time n^d with $\mu(T) < n^{-c}$. The values of parameters in this theorem are chosen so that the Majority Principle holds: for any candidate μ the fraction of strings for which μ is not acceptable is less than $2^d n^{-c}$ which is negligible for large c and n. And the resources n^d, d needed to refute a candidate μ can be even smaller than resources $n^c, c\log n$ allowed to sample the candidate μ, as c can be much larger than d.

Later we will compare this result to known results on non-existence of stochastic strings. The latter exist only for $t = t' = *$.

3.2 Super-logarithmic gap between distinguishing complexity and Kolmogorov complexity

In the proof of Theorem 8 we use the notion of an elusive set, which is interesting in its own right.

▶ **Definition 9.** A language T is called *elusive* if it is decidable in polynomial time and for all c for infinitely many n the following holds. T contains at least one word of length n, however there is no probabilistic machine M with program of length at most $c \log n$ and running time at most n^c that with probability at least n^{-c} produces a string of length n from T.

We show that under PI Assumption 1 there exists an elusive set (Theorem 22). Then we prove that any elusive set has infinitely many non-stochastic strings. There is another interesting corollary from the existence of elusive sets: there are infinitely many pairs x, r with $\mathrm{CD}^{\mathrm{poly}(n)}(x|r) \ll \mathrm{C}^{\mathrm{poly}(n)}(x|r)$ (the definition of conditional distinguishing complexity and conditional Kolmogorov complexity is given in Section 2). More specifically, the following holds.

▶ **Theorem 10.** *Under Assumption 1 for some constant d for all c there are infinitely many strings x with*

$$\mathrm{CD}^{n^d}(x|r) \leqslant \mathrm{C}^{n^c}(x|r) - c \log n$$

for 98% of r's of length n^d. Here n stands for the length of x. Moreover, under Assumption 2 (see below), in the left hand side *of the last inequality, we can replace the conditional complexity by the unconditional one:*

$$\mathrm{CD}^{n^d}(x) < \mathrm{C}^{n^c}(x|r) - c \log n.$$

▶ Assumption 2. There is a set that is decidable by deterministic Turing machines in time $2^{O(n)}$ but is not decidable by Boolean circuits of size $2^{o(n)}$ for almost all n.

The existence of pairs x, r satisfying the first part of Theorem 10 is known to be equivalent to the impossibility to separate in polynomial time non-satisfiable Boolean formulas from those having the unique satisfying assignment [3]. The latter statement (denoted by $(1SAT, SAT) \notin \mathrm{P}$) follows from the assumption NP $\neq$ RP, which is weaker than Assumption 1, using Valiant and Vazirani Lemma [11].[4] For unconditional complexity, previously it was known that there are strings with $\mathrm{CD}^{n^d}(x) < \mathrm{C}^{n^c}(x) - c \log n$ under the assumption $\mathrm{FewP} \cap \mathrm{SPARSE} \nsubseteq \mathrm{P}$ [3]. Thus the first part of Theorem 10 is not new, however its second part is.

3.3 A comparison of the notions of acceptability, plausibility and optimality

▶ **Definition 11.** A probability distribution μ is called (t, ε)-*optimal for* x, if

$$\mu(x) > \varepsilon 2^{-\mathrm{C}^t(x)}.$$

The larger t, ε are, the stronger the notion of (t, ε)-optimality is. Assume that the distribution μ is (t', α)-simple for a small α. We will explain that the definition of optimality

[4] Thus if $(1SAT, SAT) \in \mathrm{P}$ then there are no elusive sets. Is the inverse true?

makes sense for values of t which are larger than some polynomial of $|x| + t'$ and does not make any sense if, conversely, t' is larger than some polynomial of $|x| + t$.

Consider the following

▶ Assumption 3. There is a set which is decidable by deterministic Turing machines in time $2^{O(n)}$ but is not decidable by deterministic Turing machines in space $2^{o(n)}$ for almost all n.

▶ **Proposition 12.** *Under Assumption 3 the following holds. There is a constant d such that for all (t, α)-simple probability distributions μ for all strings x of length n,*

$$\log \mu(x) \leqslant -\mathrm{C}^{(n+t)^d}(x) + \alpha + d \log(n + t).$$

Assume that μ is a (t, ε)-optimal (t', α)-simple hypothesis for x and $t > (n + t')^d$ where d is the constant from Proposition 12. Then $\log \mu(x)$ differs from the maximal possible value of $\log \mu'(x)$ for (t', α)-simple hypotheses μ' by at most $\alpha + \log(1/\varepsilon) + d \log(n + t')$. This fact provides some justification for the notion of optimality. Another justification for the definition is the validity of the Majority Principle:

▶ **Proposition 13.** *For some constant c for all n and all strings x of length n for all probability distributions μ we have*

$$\mu\{x \mid \mu \text{ is not } (*, \varepsilon)\text{-optimal for } x\} < \varepsilon(n + c).$$

Conversely, if t' is larger than some polynomial of $|x| + t$ then *for all strings* there is a simple optimal hypothesis (and thus the notion of optimality becomes trivial).

▶ **Proposition 14.** *There is a constant c such that for all t every string x of length n has a $((n + t)^c, c \log(n + t))$-simple $(t, 1)$-optimal hypothesis.*

Letting $T = \{x\}$ in the definition of plausibility we can see that plausibility implies optimality:

▶ **Proposition 15.** *For all strings x and for all (t, ε)-plausible hypotheses μ for x we have* $\log \mu(x) \geqslant -\mathrm{CD}^t(x) + \log \varepsilon \geqslant -\mathrm{C}^t(x) + \log \varepsilon - O(1)$.

By Proposition 12 the first inequality in this proposition implies the following

▶ **Proposition 16.** *Under Assumption 3 there is a constant d such that for every string x of length n that has a (t_1, α)-simple (t_2, ε)-plausible hypothesis we have*

$$\mathrm{C}^{(n+t_1)^d}(x) \leqslant \mathrm{CD}^{t_2}(x) + \alpha + \log(1/\varepsilon) + d \log(n + t_1).$$

Therefore strings with a large gap between distinguishing complexity and Kolmogorov complexity do not have simple plausible explanations. From the result of [3] cited above it follows that (under some complexity theoretic assumptions) for some d for every c there are infinitely many strings x without $(n^c, c \log n)$-simple (n^d, n^{-c})-plausible hypotheses (where n denotes the length of x).

Thus plausibility implies acceptability and optimality. Is there any implication in the reverse direction? Assuming the existence of a Pseudo Random Number Generator $G : \{0,1\}^n \to \{0,1\}^{2n}$ we can show that acceptability does not imply neither plausibility, nor optimality.

▶ Assumption 4 (Existence of PRNG). There is a polynomial time computable function $G : \{0,1\}^* \to \{0,1\}^*$, such that $|G(s)| = 2|s|$ and for every sequence of Boolean circuits $\{C_n\}$ with $2n$ inputs and 1 output such that the size of C_n is bounded by a polynomial of n, the difference of probabilities of events $C_n(G(s)) = 1$ and $C_n(r) = 1$ tends to 0 faster than every inverse polynomial of n (that is, for any polynomial p for all sufficiently large n the difference of probabilities is less than $1/p(n)$). We assume here the uniform distributions over strings s and r of length n and $2n$, respectively.

▶ **Theorem 17.** *Assume that there is PRNG $G : \{0,1\}^n \to \{0,1\}^{2n}$, as in Assumption 4. Then for all c for all sufficiently large n for 99% of strings s of length n the uniform distribution over strings of length $2n$ is a $(n^c, c\log n, n^{-c}/200)$-acceptable hypothesis for $G_n(s)$.*

▶ Remark. Note that the uniform distribution is neither optimal ($\mathrm{C}^{\mathrm{poly}(n)}(x) \leqslant n + O(1)$, and $\log \mu(x) = -2n$), nor plausible (recall Example 2) hypothesis for x. By counting arguments for almost all s for $x = G_n(s)$ it holds $\mathrm{CD}^{\mathrm{poly}(n)}(x) \approx \mathrm{C}^{\mathrm{poly}(n)}(x) \approx \mathrm{C}(x) \approx n$. Therefore, there are strings satisfying Theorem 17 and having the latter property.

Finally, for simple hypotheses μ and for strings with $\mathrm{CD}^{\mathrm{poly}}(x) \approx \mathrm{C}^{\mathrm{poly}}(x)$ optimality implies plausibility and hence acceptability. However this time we need that μ can be computed rather than sampled in a short time by a short program.

▶ **Theorem 18.** *Under Assumption 2 there is a constant c such that the following holds true. Let μ be a probability distribution μ such that the function $x \mapsto \mu(x)$ can be computed by a program of length α in time t. Assume further that $\mu(x) > \varepsilon 2^{-\mathrm{CD}^{(n+t+t_1)^c}(x)}$, where n is the length of x and t_1 an arbitrary number. Then μ is a $(t_1, \varepsilon 2^{-\alpha - c\log n})$-plausible hypothesis for x.*

Notice that in this theorem instead of $((n+t+t_1)^c, \varepsilon)$-optimality we use a stronger condition $\mu(x) > \varepsilon 2^{-\mathrm{CD}^{(n+t+t_1)^c}(x)}$ (with distinguishing complexity in place of Kolmogorov complexity). However for strings x and t_2 with $\mathrm{C}^{t_2}(x) \leqslant \mathrm{CD}^{(n+t+t_1)^c}(x) + \beta$ we can replace that condition by the condition of $(t_2, \varepsilon 2^{\beta})$-optimality of μ for x. Informally speaking, if $\mathrm{C}^{\mathrm{poly}}(x) \approx \mathrm{CD}^{\mathrm{poly}}(x)$ then optimality for x implies plausibility for x.

3.4 Non-stochastic strings in classical Algorithmic Statistics

In Algorithmic Statistics without resource bounds [4, 6, 7, 8, 12, 13] plausibility of a statistical hypothesis μ for x is measured by one parameter $-\log \mu(x) - \mathrm{C}(x|\mu)$, called *randomness deficiency of x w.r.t. μ*. Probability distributions can be represented by the lists of pairs (a string, its probability) ordered in a standard way. Thus we can talk on conditional Kolmogorov complexity $\mathrm{C}(x|\mu)$ and of Kolmogorov complexity $\mathrm{C}(\mu)$ of μ itself. Up to an additive constant $\mathrm{C}(\mu)$ coincides with the length of the shortest program sampling μ (assuming that the program always halts).

Neglibibility of randomness deficiency is similar to all three our definitions of a good hypothesis. More specifically the inequality $-\log \mu(x) - \mathrm{C}(x|\mu) < \beta$ is similar to saying that μ is $(*, \alpha, 2^{-\beta})$-acceptable, $(*, 2^{-\beta})$-plausible and $(*, \alpha, 2^{-\beta})$-optimal for x. However there is an important difference. The inequality $-\log \mu(x) - \mathrm{C}(x|\mu) < \gamma$ implies that μ is $(*, \gamma + O(1))$-optimal for x, but not the other way around. If $-\log \mu(x) - \mathrm{C}(x|\mu) < \gamma$ then for every set $T \ni x$ *accepted by a non-deterministic* program p we have $\mu(T) > 2^{-|p|-\gamma}$. Conversely, if $-\log \mu(x) - \mathrm{C}(x|\mu) \geqslant \gamma$, then there is a set $T \ni x$ *accepted by a short non-deterministic* program (of length about $\mathrm{C}(\mu)$) with $\mu(T) \leqslant 2^{-\gamma}$.

In contrast, both the notion of $(*, \gamma)$-plausibility and the notion of $(*, \alpha, \varepsilon)$-acceptability are defined by means of *deterministic* recognizing machines. Thus $-\log \mu(x) - \mathrm{C}(x|\mu) < \gamma$ implies $(*, \gamma)$-plausibility but not the other way around (with logarithmic accuracy: the inequality $-\log \mu(x) - \mathrm{C}(x|\mu) < \gamma$ implies only $(*, \gamma + O(\log n))$-plausibility.)

A string x is called *Kolmogorov* (α, β)*-stochastic* if there is a probability distribution μ with $\mathrm{C}(\mu) \leqslant \alpha$ and $-\log \mu(x) - \mathrm{C}(x|\mu) \leqslant \beta$. As we have just explained, Kolmogorov (α, β)-stochasticity implies the existence of a $(*, \alpha)$-simple $(*, 2^{-\beta})$-plausible (and hence $(*, \alpha_2, 2^{-\beta-\alpha_2})$-acceptable for all α_2) hypothesis. (Again we ignore logarithmic terms.)

Shen proved the existence of Kolmogorov (α, β)-non-stochastic string for α, β that are linear in n:

▶ **Theorem 19** ([10]). *For some constant c for all n and all α, β, with $2\alpha + \beta < n - c \log n$, there is a Kolmogorov (α, β)-non-stochastic string of length n.*

As we have mentioned, this statement does not imply the existence of non-stochastic strings in our sense (even for very large values of time parameters). However the techniques of [10] can be used to prove the following:

▶ **Theorem 20.** *For all n and all α, β with $\alpha + \beta < n$ there is a string of length n that has no $(*, \alpha)$-simple $(*, \alpha + O(\log n), 2^{-\beta})$-acceptable hypotheses.*

It is not hard to see that Theorem 20 implies Theorem 19. Later the term 2α in Theorem 19 was replaced with α:

▶ **Theorem 21** ([13]). *For some constant c for all n and all α, β, with $\alpha + \beta < n - c \log n$, there is a Kolmogorov (α, β)-non-stochastic string of length n.*

This result is optimal up to logarithmic terms. Indeed, for all x of length n and all $\alpha \leqslant n$ the uniform distribution μ over strings of length n that have the same α first bits as x, has complexity about α and randomness deficiency at most $n - \alpha$:

$$-\log \mu(x) - \mathrm{C}(x|\mu) = n - \alpha - \mathrm{C}(x|\mu) \leqslant n - \alpha.$$

So using the known methods we can show the existence of strings of length n that have no $(*, \alpha_1)$-simple $(*, \alpha_2, \varepsilon)$-acceptable hypotheses for $\alpha_1, \log(1/\varepsilon) = \Omega(n)$ and for α_2 which are only logarithmically larger than α_1. It is essential for those methods that the running time can be arbitrary large and hence they cannot be used in the case when the running time is bounded by a polynomial of the length.

The notion of an optimal hypothesis is also borrowed from the classical Algorithmic Statistics. A distribution μ with small Kolmogorov complexity is called optimal if $\log \mu(x)$ is close to $-\mathrm{C}(x)$, which is equivalent to saying that the randomness deficiency is small. However, optimality was studied also for distribution μ with large Kolmogorov complexity, in which case optimality was defined as $\log \mu(x) \approx \mathrm{C}(\mu) - \mathrm{C}(x)$. Using the Symmetry of Information, we can show that the randomness deficiency never exceeds the 'optimality deficiency' $\mathrm{C}(\mu) - \mathrm{C}(x) - \log \mu(x)$, but not the other way around [13]. However in the definition of Kolmogorov stochasticity, we can use the optimality deficiency instead of randomness deficiency: for a string of length n there is an $*, \alpha$-simple hypothesis with optimality deficiency less than β if and only if the string is Kolmogorov α, β-stochastic. More accurately, both directions 'if' and 'only if' hold up to adding some terms of order $O(\log n)$ to parameters α, β [13].

4 Open questions

▶ **Question 1.** *Under which other assumptions (different from Assumption 1) there are non-stochastic strings and elusive sets? Under which other assumptions (different* $(1SAT, SAT) \in \mathrm{P}$ *and* $\mathrm{P} = \mathrm{PSPACE}$*) there are no elusive sets and all strings are stochastic?*

▶ **Question 2.** *Let us replace in the definitions of a plausible and acceptable hypothesis deterministic machines by non-deterministic ones. Do the notions of a plausible and acceptable hypothesis and of stochastic string become stronger?*

▶ **Question 3.** *Are there strings that do not possess simple optimal hypotheses?*

▶ **Question 4.** *How acceptability is related to optimality for strings* x *with* $\mathrm{CD}^{poly}(x) \ll \mathrm{C}^{poly}(x)$*?*

▶ **Question 5.** *Are there non-stochastic strings with polynomial bounds for time and linear bounds for program length: is it true that for some* c *and* $\varepsilon < 1$ *for all* d *and all* $\delta < 1$ *for all but finitely many* n *every string* x *of length* n *has an* $(n^c, \varepsilon n)$*-simple* $(n^d, \delta n, n^{-c})$*-acceptable hypothesis?*

5 The proofs

5.1 Proof of Proposition 7

The first inequality: the number of sets recognized by a program of length less than α is less than 2^α and each such set contains a fraction at most ε of all n-bit strings w.r.t. μ.

The second inequality: w.l.o.g. we may consider only sets T recognized by programs of length less than $n + c$ (for some constant c). Indeed, assume that a set $T \ni x$ witnesses implausibility of μ for x and is recognized by a program of length $l > n + c$. Then $\mu(x) \leqslant \mu(T) \leqslant \varepsilon 2^{-n-c}$. Thus the set $\{x\}$, whose complexity is less than $n + c$, witnesses implausibility of μ for x (if c is large enough). Then we can repeat the arguments from the previous paragraph: for every fixed l any set T recognized by a program of length l refutes a fraction at most $\varepsilon 2^{-l}$ of all strings and the number of programs of length l is 2^l, thus all together they refute a fraction at most ε of strings of length n.

5.2 Proof of Theorem 8

▶ **Theorem 22.** *Under Assumption 1 there exists an elusive set.*

Proof. Fix a language L over the unary alphabet $\{1\}$ satisfying Assumption 1. Since $L \in \mathrm{NP}$, it can be represented in the form

$$L = \{1^k \mid \exists x \in \{0,1\}^{k^c} \ R(1^k, x)\},$$

where $c > 0$ is a natural number and R a relation decidable in time $\mathrm{poly}(k)$.

Consider the set

$$T = \{x \in \{0,1\}^{k^c} \mid R(1^k, x)\}.$$

Obviously T can be recognized in polynomial time. Let us show that T is elusive.

Let d be any constant. For the sake of contradiction assume that for some m for all $k > m$ with $1^k \in L$ there is a program M_k of length $d \log k^c$ that, with probability at least k^{-cd}, in time k^{cd} prints a string from T of length k^c. To obtain a contradiction we construct the following probabilistic algorithm that finds in polynomial time with failure probability at most $1/2$ a certificate for membership of an input string 1^k in L:

The algorithm. On input 1^k we run all randomized programs of length $d \log k^c$ in k^{cd} steps. Each program is run k^{cd} times. For each string x output by any of those programs we check the equality $R(1^k, x) = 1$. If the equality holds true for at least one of those x's, we output any such x. (The end of the Algorithm.)

This algorithm runs in polynomial time. Let us bound the probability of failure on any input $1^k \in L$. We assume that for all $1^k \in L$ with $k > m$ there is a randomized program of length $d \log k^c$ that produces a string from T with probability at least k^{-cd}. The probability that k^{cd} times its output falls outside T is less than $(1 - k^{-cd})^{k^{cd}} \leqslant 1/e$. Therefore for all $k > m$ the algorithm fails with probability at most $1/e$, which is a contradiction. ◀

Proof of Theorem 8. By Theorem 22 there is an elusive set T. For some d there is a machine with program of length at most d recognizing T in time n^d.

Let $T^{=n}$ denote the set of all strings of length n from T. For every n such that $T^{=n} \neq \emptyset$ pick any string x_n from $T^{=n}$. We claim that for any constant c for infinitely many n the string x_n does not have $(n^c, c \log n)$-simple (n^d, d, n^{-c})-acceptable hypotheses.

For the sake of contradiction assume that for some m for all $n > m$ there is such hypothesis μ_n. As $x_n \in T$ and T is recognized by a program of length at most d in time n^d we have $\mu(T) \geqslant n^{-c}$. Thus for each such n the probabilistic program of length less than $c \log n$ sampling the distribution μ_n in time n^c produces a string from T with probability at least n^{-c}, which contradicts the assumption that T is elusive. ◀

5.3 Proof of Theorem 10

▶ **Proposition 23.** *Assume that L is an elusive set. Then for all constants c there is a constant d such that there are infinitely many $x \in L$ with*

$$\mathrm{CD}^{n^d}(x|r) \leqslant \mathrm{C}^{n^c}(x|r) - c \log n$$

for 99% of strings r of length n^d. Here n denotes the length of x.

This proposition follows from Sipser's lemma.

▶ **Lemma 24** ([9])**.** *For every language L recognizable in polynomial time there is a constant d such that for all n for 99% of strings r of length n^d and all $x \in L^{=n}$ we have*

$$\mathrm{CD}^{n^d}(x|r) \leqslant \log |L^{=n}| + d \log n.$$

Proof Proposition 23. Let d be the constant from Sipser's Lemma applied to the given elusive language L. Let c be an arbitrary constant. By Sipser's lemma it suffices to show that $\log |L^{=n}| + d \log n$ is less than the right hand side of the inequality we have to prove. More precisely, we have to show that

$$\log |L^{=n}| + d \log n \leqslant \mathrm{C}^{n^c}(x|r) - c \log n$$

for infinitely many $x \in L$ and for 99% of r of length n^d.

For the sake of contradiction assume that for some m for all $n > m$ for all $x \in L^{=n}$ we have

$$\mathrm{C}^{n^c}(x|r) < \log |L^{=n}| + (c + d) \log n$$

for at least 1% of r's. For any such n consider the program M_n of probabilistic machine that samples a random string w of length less that $\log |L^{=n}| + (c + d) \log n$ (all such strings are

equiprobable) and a string r of length n^d. Then M_n considers w as a program of a string conditional to r, runs that program in n^c steps and outputs its result (if any). Thus M_n outputs every $x \in L^{=n}$ with probability at least $1/(100|L^{=n}|n^{c+d})$. And hence for all $n > m$ with non-empty $L^{=n}$ the output M_n falls in $L^{=n}$ with probability at least

$$|L^{=n}|/(100|L^{=n}|n^{c+d}) = 1/100n^{c+d}.$$

This contradicts the assumption that L is an elusive set, as M_n runs in time poly(n) and its program length is $O(\log n)$. ◀

The first part Theorem 10 follows immediately from Theorem 22 and Proposition 23. Let us prove the second part of Theorem 10. In [14], it was shown that under Assumption 2 we can replace in Sipser's lemma conditional complexity by the unconditional one.

▶ **Theorem 25** (Theorem 3.2 in [14]). *Under Assumption 2 for all $L \in$ PSPACE/poly there is a constant d such that for all $x \in L^{=n}$ we have*

$$\mathrm{CD}^{n^d, L^{=n}}(x) \leqslant \log |L^{=n}| + d \log n.$$

Moreover the constant d depends only on the length of the advice string for L and on the space bound for L.

In the notation $\mathrm{CD}^{n^d, L^{=n}}(x)$ the superscript $L^{=n}$ means that the distinguishing program is granted the access to an oracle for $L^{=n}$. If L is decidable on polynomial space we can drop this superscript.

Combining this theorem with the proof of Proposition 23 and Theorem 22 we obtain the proof of the second part of Theorem 10.

▶ Remark. In [3], a weaker result is derived from an assumption that is not comparable with our one:

▶ **Theorem 26** ([3]). *Assume that* FewP $\cap$ SPARSE $\not\subseteq$ P. *Then for some constant d for all c for infinitely many x we have*

$$\mathrm{CD}^{n^d}(x) < \mathrm{C}^{n^c}(x) - c \log n.$$

Here n denotes the length of x.

▶ Remark. In [3], the following relation between (1 SAT, SAT) and distinguishing complexity was discovered:

▶ **Theorem 27** ([3]). *The following are equivalent:*

(1) $(1SAT, SAT) \notin$ P.

(2) *For some d for all c there are x and y with*

$$\mathrm{CD}^{(|x|+|y|)^d}(x|y) \leqslant \mathrm{C}^{(|x|+|y|)^c}(x|y) - c \log(|x| + |y|).$$

From Theorem 27 and Theorem 10 we obtain the following implication NE $\not\subseteq$ RE $\Rightarrow$ $(1SAT, SAT) \notin$ P, which is not surprising since $(1SAT, SAT) \in$ P implies NP = RP using the Valiant–Vazirani Lemma.

5.4 Proof of Proposition 12

▶ **Definition 28.** A probability distribution σ over $\{0,1\}^*$ is called P-samplable, if there is a program of randomized machine that samples this distribution in time bounded by a polynomial of the length of the output.

▶ **Theorem 29** (Lemma 3.2 in [1]). *Under Assumption 3 for every* P*-samplable probability distribution σ there is a d such that for all x of length n,*

$$\mathrm{C}^{n^d}(x) \leqslant -\log\sigma(x) + d\log n.$$

Proof of Proposition 12. Assume that μ is sampled by a program q of length less than α in time t. Assume that $\alpha < n$ as otherwise the statement is obvious (the complexity of x with a polynomial time bound does not exceed its length).

Consider the following P-samplable probability distribution σ: we choose a random t with probability proportional to $1/t^2$, then we choose a random program q' of a randomized machine with probability proportional to $2^{-|q'|}/|q'|^2$, run that program in t steps and output the triple $(1^t, q', x)$, where x is the result of q' (if any, and the empty string otherwise). The triple $(1^t, q', x)$ is encoded in such a way that the code length be polynomial in $t + |q'| + |x|$. By Theorem 29

$$\mathrm{C}^{|y|^d}(y) \leqslant -\log\sigma(y) + d\log|y|$$

for some constant d and all y. Letting $y = (1^t, q, x)$, we obtain

$$\begin{aligned}\mathrm{C}^{|(1^t,q,x)|^d}(1^t, q, x) &\leqslant -\log\sigma(1^t, q, x) + c\log|(1^t, q, x)| \\ &\leqslant 2\log t + \alpha + 2\log\alpha - \log\mu(x) + O(\log(t + |x|)).\end{aligned}$$

Since the complexity of any entry of a tuple does not exceed the complexity of the tuple itself, we get the sought inequality. ◀

5.5 Proof of Proposition 13

Indeed, if μ is not $*, \beta$-optimal for x, then $\mu(x) < 2^{-\beta-\mathrm{C}(x)}$. The sum of probabilities of all such words is less than ε times the sum of $2^{-\mathrm{C}(x)}$ over all x of length n. The latter sum is less than $n + O(1)$, since $\mathrm{C}(x) \leqslant n + O(1)$ for all x of length n and for all fixed k the sum of $2^{-\mathrm{C}(x)}$ over all x with $\mathrm{C}(x) = k$ is at most 1 (there are at most 2^k such x's).

5.6 Proof of Proposition 14

Consider the machine that chooses a random program of length $\mathrm{C}^t(x)$ (with uniform distribution), runs it in t steps and outputs its result (if any). The program of this machine has length $O(\log(n + t))$ and its running time is bounded by a polynomial in $t + n$. With probability at least $2^{-\mathrm{C}^t(x)}$ that machine prints x hence it samples a probability distribution μ that is $\mathrm{poly}(n + t), 1$-optimal for x.

5.7 Proof of Theorem 17

Fix an arbitrary constant c. For any set T_n of strings of length $2n$ recognizable by a program of length less than $c\log n$ in time n^c we can construct a Boolean circuit C_n recognizing that set whose size is bounded by a polynomial of n (that polynomial depends only on c). Therefore there is a function $\varepsilon(n)$ that tends to 0 faster than any inverse polynomial of n

and such that the probabilities of events $G_n(s) \in T_n$ and $r \in T_n$ differ at most by $\varepsilon(n)$ for any such set T_n.

For the sake of a contradiction assume that for infinitely many n for 1% of strings s of length n there is a set T_s recognizable by a program of length less than $c \log n$ in time n^c with $\Pr[r \in T_s] < n^{-c}/200$. Consider the union $\mathcal{T}_n$ of all such sets. Since $G_n(s) \in \mathcal{T}_n$ for all such s, the probability of event $G_n(s) \in \mathcal{T}_n$ is at least $1/100$. On the other hand, the probability of event $r \in \mathcal{T}_n$ is less than $2^{c \log n}$ (the number of sets T_s) times $n^{-c}/200$, which equals $1/200$. Thus the difference of probabilities of events $r \in \mathcal{T}_n$ and $G_n(s) \in \mathcal{T}_n$ is greater than $1/200$.

Recall now that for each n probabilities of events $G_n(s) \in T_s$ and $r \in T_s$ differ by at most $\varepsilon(n)$, which tends to 0 faster than any inverse polynomial of n. The number of T_s is less than $2^{c \log n}$. Thus we obtain the inequality $2^{c \log n}\varepsilon(n) > 1/200$ for infinitely many n, which is a contradiction.

5.8 Proof of Theorem 18

First we derive a corollary from Theorem 25.

▶ **Corollary 30.** *Under Assumption 2 for some constant d for all n for every program q of length at most $3n$ that recognizes a set $T \subset \{0,1\}^n$ in time t, for all $x \in T$ we have*

$$\mathrm{CD}^{(n+t)^d}(x) \leqslant \log |T| + |q| + d \log(n+t).$$

Proof. Fix any sequence of strings q_n with $q_n \leqslant 3n$. Let

$$L = \bigcup_{n,t} T_{n,t}, \text{ where } T_{n,t} = \{0^{[n,t]-n}1x \mid |x| = n,\ q_n(x) = 1 \text{ in time } t\}.$$

Here $[n,t]$ – denotes a polynomial computing a 1-1-mapping from pairs of natural numbers to natural numbers such that $[n,t] \geqslant n,t$. Given the length of any word from $T_{n,t}$ we can compute n and t in polynomial time. Therefore $L \in \mathrm{P}/3n$ and $L^{=([n,t]+1)} = T_{n,t}$. Hence we can apply Theorem 25 to L and conclude that

$$\mathrm{CD}^{([n,t]+1)^d, q_n}(0^{[n,t]-n}1x) \leqslant \log |T_{n,t}| + d \log(n+t)$$

for all t, for all x of length n and all sequences $\{q_n\}$ as above. The constant d does not depend on $\{q_n\}$, therefore this inequality holds for all n,t, for all x of length n and all $q \leqslant 3n$. Plug into this inequality t, q, n, x from the conditions of theorem. We obtain

$$\mathrm{CD}^{([n,t]+1)^d, q}(0^{[n,t]-n}1x) \leqslant \log |T| + d \log(n+t).$$

It remains to append to the program of this length distinguishing $0^{[n,t]-n}1x$ from other strings the information about n,t and q. In this way we get a distinguishing program for x of length $\log |T| + |q| + O(\log(n+t))$ with running time $\mathrm{poly}(n,t)$ that does not need an oracle for T. ◀

Proof of Theorem 18. Assume the contrary: there is $T \ni x$ recognizable by a program of length l in time t_1 with $\mu(T) < \varepsilon 2^{-l-\alpha-c\log n}$ (where the constant c will be chosen later). Then consider the set $T' = \{x' \in T \mid \mu(x') \geqslant 2^{-i}\}$ where $-i$ stands for the integer part of the binary logarithm of $\mu(x)$. This set has at most $\mu(T)2^i \leqslant 2\mu(T)/\mu(x)$ strings (of length n) and can be recognized in time $t + t_1$ by a program of length $\alpha + l + O(\log\log(1/\mu(x)))$.

W.l.o.g. we may assume that $\mu(x) \geqslant 2^{-n}$ and that $\alpha, l \leqslant n$. Thus the length of that program is less than $3n$. Corollary 30 implies that for some constant d

$$\mathrm{CD}^{(n+t+t_1)^d}(x) \leqslant \log \mu(T) - \log \mu(x) + \alpha + l + d \log n,$$

that is,

$$\begin{aligned}\log \mu(x) &\leqslant -\mathrm{CD}^{(n+t+t_1)^d}(x) + \alpha + l + \log \mu(T) + d \log n \\ &\leqslant -\mathrm{CD}^{(n+t+t_1)^d}(x) + \alpha + l + \log \varepsilon - l - \alpha - c \log n + d \log n \\ &= g\,\mathrm{CD}^{(n+t+t_1)^d}(x) + \log \varepsilon - c \log n + d \log n.\end{aligned}$$

Let $c = d + 1$. Then the last inequality implies that

$$\log \mu(x) < -\mathrm{CD}^{(n+t+t_1)^d}(x) + \log \varepsilon \leqslant -\mathrm{CD}^{(n+t+t_1)^c}(x) + \log \varepsilon,$$

which contradicts the condition of the theorem. ◀

Proof of Theorems 20 and 19

Proof of Theorem 20. For every μ sampled by a program of length $< \alpha$ consider the set of all x' satisfying the inequality $\mu(x') \geqslant 2^{-\beta}$. For any fixed μ there at most 2^β such x' (otherwise the sum of their probabilities would exceed 1). Therefore the total number of strings in all such sets is less than

$$2^\alpha 2^\beta < 2^n.$$

Here the first factor is an upper bound for the number of μ and the second factor the number of x' for a fixed μ.

Let x by the lex first string of length n outside all such sets. Its Kolmogorov complexity is at most $\alpha + O(\log n)$, as we can find it from the number N of distributions μ sampled by a program of length $< \alpha$ and from parameters α, β (from α and N we can find all such distributions by running in parallel all programs of length less than α until we find N distributions; then for every of the distributions μ we can find the set of strings x' with $\mu(x') \geqslant 2^{-\beta}$).

Let us show that x does not possess $(*, \alpha)$-simple $(*, \alpha + O(\log n), 2^{-\beta})$-acceptable hypotheses. Assume that μ is sampled by a program of length $< \alpha$. Consider the set $T = \{x\}$. Its complexity is at most $\alpha + O(\log n)$ and μ-probability is less than $2^{-\beta}$ by construction. Hence the set T witnesses that μ is not acceptable for x. ◀

Proof of Theorem 19. Assume that α', β' satisfy the inequality $2\alpha' + \beta' + c \log n < n$ where c is the constant hidden in the O-notation in Theorem 20 (actually a little larger). Let in Theorem 20 $\alpha = \alpha'$ and $\beta = \beta' + \alpha' + c \log n$. If the word x existing by Theorem 20 were Kolmogorov (α', β')-stochastic, then it would have $(*, \alpha')$-simple $(*, \alpha_2, 2^{-\beta'-\alpha_2})$-acceptable hypothesis for all α_2. Letting $\alpha_2 = \alpha + c \log n$, we would derive that x has an $(*, \alpha)$-simple $(*, \alpha + c \log n, 2^{-\beta})$-acceptable hypothesis, which contradicts the statement of Theorem 20. ◀

6 Non-stochastic strings and P = PSPACE

In this section we show why we need some complexity-theoretic assumption in Theorem 8 – its statement implies P ≠ PSPACE (Theorem 31). In other words, P = PSPACE implies that the conclusion in Theorem 8 is false. However, this is due to the fact that, in Theorem 8, the

length of a program (and its running time) recognizing the refutation set T does not depend on program length and time to sample the distribution. If we allow the former depend on the latter then, on the contrary, P = PSPACE implies that non-stochastic strings exist (Theorem 32).

▶ **Theorem 31.** *Assume that* P = PSPACE. *Then for every c there is d for which every string of length n has an $(n^d, 2\log n)$-simple $(n^c, c\log n, n^{-d})$-acceptable hypothesis.*

Proof. Fix a constant c. Define sets $A_0, A_1, \dots$ recursively: $A_0 = \{0,1,\}^n$ and for $i > 0$ let

$$A_i = \{x \in A_{i-1} \mid \exists\ (n^c, c\log n, n^{-c})\text{-simple } T \ni x\ |T \cap A_{i-1}| \leqslant 2^n n^{-i-c}\}.$$

The definition of A_i implies that it has at most $2^n n^{-i}$ words. Indeed, there are less than n^c $(n^c, c\log n)$-simple sets and each of them contributes at most $2^n n^{-i-c}$ strings to A_i.

Thus A_n is empty. On the other hand, $A_0 = \{0,1\}^n$, therefore for every string x of length n there is $i \leqslant n$ with $x \in A_i \setminus A_{i+1}$. For a given x fix such i and consider the distribution μ_i sampled as follows. Sample a random $j \leqslant 2^n n^{-i}$ and output jth in the lexicographical order word from A_i (if there is no such word, then the last one, say).

Assume that there is an $(n^c, c\log n)$-simple $T \ni x$ with $\mu_i(T) < n^{-d}$. The probability of each string from A_i is at least $2^{-n} n^i$ hence we have

$$|A_i \cap T| < n^{-d} 2^n n^{-i} \leqslant 2^n n^{-(i+1)-c}$$

(the last inequality holds provided $d \geqslant c+1$). Hence $x \in A_{i+1}$, which contradicts to the choice of i.

It remains to show that μ_i is $(n^d, 2\log n)$-simple provided d is large enough. The distribution μ_i can be identified by numbers n, i, hence there is a program of length $2\log n$ sampling μ_i. Given the index of a string x in A_i we can find x on the space polynomial in n and n^c. Under assumption P = PSPACE, we can do it in time polynomial in n and n^c and hence μ_i is $(n^d, 2\log n)$-simple for some d. ◀

In this theorem the time n^d to sample an $(n^c, c\log n, n^{-d})$-acceptable hypothesis μ for x can be much larger than the time n^c allowed to refute μ. Does a similar statement hold for d that does not depend on c? The next theorem answers this question in the negative.

▶ **Theorem 32.** *Assume that* P = PSPACE. *Then for some constant e for every n, α, t there is a string of length n that has no (t,α)-simple $((\alpha+t+n)^e, \alpha+2\log t+2\log n, 2^{-n+\alpha})$-acceptable statistical hypotheses.*

Plugging $t = n^d$ and $\alpha = d\log n$ we get, for each n, a string of length n with no $(n^d, d\log n)$-simple $(n^{ed}, (2d+2)\log n, 2^{-n+d\log n})$-acceptable hypotheses. Thus for any d for some $c = O(d)$ there are infinitely many strings which have no $(n^d, 2\log n)$-simple $(n^c, c\log n, n^{-d})$-acceptable hypotheses.

Proof. Let μ_p^t denote the probability distribution sampled by a program p in time t. Consider the arithmetic mean of all (t,α)-simple distributions: $\mu(x) = 2^{-\alpha}\sum_{|p|<\alpha} \mu_p^t(x)$. Let x stand for the lex first string of length n such that $\mu(x) \leqslant 2^{-n}$. This string can be found on space $\mathrm{poly}(n+t+\alpha)$. Using the assumption P = PSPACE we conclude that x can be found in time $p(n+t+\alpha)$ from t, α, n, where p is a polynomial.

We claim that x has has no (t,α)-simple $(p(\alpha+t+n), \alpha+2\log t+2\log n, 2^{-n+\alpha})$-acceptable statistical hypotheses. For the sake of contradiction assume that μ_p^t is such a hypothesis for x. By construction we have $\mu_p^t(x) \leqslant 2^{-n+\alpha}$ and hence the singleton set $T = \{x\}$ has small probability. It can be recognized in time $p(\alpha+t+n)$ by a program of length less than $\alpha + 2\log t + 2\log n$, consisting of t and n in the self-delimiting encoding followed by p. ◀

▶ Remark. Assume that, in the definitions of a simple and acceptable hypothesis, we would restrict space instead of time. Then in Theorems 31 and 32 we would not need any assumptions.

References

1. L. Antunes and L. Fortnow, Worst-Case Running Times for Average-Case Algorithms. In *Proceedings of the 24th IEEE Conference on Computational Complexity*, pages 298–303. IEEE, 2009.
2. H. Buhrman, L. Fortnow, and S. Laplante, Resource-Bounded Kolmogorov Complexity Revisited. *SIAM Journal on Computing*, 31(3):887–905. 2002.
3. L. Fortnow and M. Kummer, On resource-bounded instance complexity. *Theoretical Computer Science*, vol. 161, issues 1–2, pages 123–140, 1996.
4. P. Gács, J. Tromp, and P. M. B. Vitányi, Algorithmic statistics. *IEEE Transactions on Information Theory*, v. 47, no. 6, pages 2443–2463, 2001.
5. R. Impagliazzo and G. Tardos, Decision versus search problems in super-polynomial time. In *Proceedings of the 30th Annual Symposium on Foundations of Computer Science*, pages 222–227, 1989.
6. A. N. Kolmogorov, The complexity of algorithms and the objective definition of randomness. Summary of the talk presented April 16, 1974 at Moscow Mathematical Society. *Uspekhi matematicheskikh nauk*, Russia, 29(4[178]), 155, 1974.
7. M. Li and P. M. B. Vitányi, *An Introduction to Kolmogorov Complexity and its Applications*. 3rd ed., Springer, New York, 2008.
8. A. Shen, V. Uspensky, and N. Vereshchagin, *Kolmogorov complexity and algorithmic randomness*. MCCME, 2013 (Russian). English translation: URL: `http://www.lirmm.fr/~ashen/kolmbook-eng.pdf`
9. M. Sipser, A complexity theoretic approach to randomness. In *Proceedings of the 15th ACM Symposium on the Theory of Computing*, pages 330–335, 1983.
10. A. Shen, The concept of (α, β)-stochasticity in the Kolmogorov sense, and its properties. *Soviet Math. Dokl.*, v. 28, no. 1, pages 295–299, 1983.
11. L. Valiant and V. Vazirani, NP is as easy as detecting unique solutions. *Theoretical Computer Science*, 47:85–93, 1986.
12. Nikolay K. Vereshchagin and Alexander Shen, Algorithmic Statistics: Forty Years Later. *Computability and Complexity* 2017:669–737, 2017.
13. N. K. Vereshchagin and P. M. B. Vitányi, Kolmogorov's structure functions with an application to the foundations of model selection, *IEEE Transactions on Information Theory*, vol. 50, no. 12, pages. 3265–3290, 2004.
14. N. V. Vinodchandran and M. Zimand, On Optimal Language Compression for Sets in PSPACE/poly. *Theory of Computing Systems*, 56:581, 2015.

Conspiracies Between Learning Algorithms, Circuit Lower Bounds, and Pseudorandomness*

Igor C. Oliveira[1] and Rahul Santhanam[2]

1 **Faculty of Mathematics and Physics, Charles University in Prague, Prague, Czech Republic**
igor.oliveira@karlin.mff.cuni.cz

2 **Department of Computer Science, University of Oxford, Oxford, UK**
rahul.santhanam@cs.ox.ac.uk

Abstract

We prove several results giving new and stronger connections between learning theory, circuit complexity and pseudorandomness. Let $\mathfrak{C}$ be any typical class of Boolean circuits, and $\mathfrak{C}[s(n)]$ denote n-variable $\mathfrak{C}$-circuits of size $\leq s(n)$. We show:

Learning Speedups. If $\mathfrak{C}[\mathsf{poly}(n)]$ admits a randomized weak learning algorithm under the uniform distribution with membership queries that runs in time $2^n/n^{\omega(1)}$, then for every $k \geq 1$ and $\varepsilon > 0$ the class $\mathfrak{C}[n^k]$ can be learned to high accuracy in time $O(2^{n^\varepsilon})$. There is $\varepsilon > 0$ such that $\mathfrak{C}[2^{n^\varepsilon}]$ can be learned in time $2^n/n^{\omega(1)}$ if and only if $\mathfrak{C}[\mathsf{poly}(n)]$ can be learned in time $2^{(\log n)^{O(1)}}$.

Equivalences between Learning Models. We use learning speedups to obtain equivalences between various randomized learning and compression models, including sub-exponential time learning with membership queries, sub-exponential time learning with membership and equivalence queries, probabilistic function compression and probabilistic average-case function compression.

A Dichotomy between Learnability and Pseudorandomness. In the non-uniform setting, there is non-trivial learning for $\mathfrak{C}[\mathsf{poly}(n)]$ if and only if there are no exponentially secure pseudorandom functions computable in $\mathfrak{C}[\mathsf{poly}(n)]$.

Lower Bounds from Nontrivial Learning. If for each $k \geq 1$, (depth-d)-$\mathfrak{C}[n^k]$ admits a randomized weak learning algorithm with membership queries under the uniform distribution that runs in time $2^n/n^{\omega(1)}$, then for each $k \geq 1$, $\mathsf{BPE} \not\subseteq$ (depth-d)-$\mathfrak{C}[n^k]$. If for some $\varepsilon > 0$ there are P-natural proofs useful against $\mathfrak{C}[2^{n^\varepsilon}]$, then $\mathsf{ZPEXP} \not\subseteq \mathfrak{C}[\mathsf{poly}(n)]$.

Karp-Lipton Theorems for Probabilistic Classes. If there is a $k > 0$ such that $\mathsf{BPE} \subseteq \mathtt{i.o.}\mathsf{Circuit}[n^k]$, then $\mathsf{BPEXP} \subseteq \mathtt{i.o.}\mathsf{EXP}/O(\log n)$. If $\mathsf{ZPEXP} \subseteq \mathtt{i.o.}\mathsf{Circuit}[2^{n/3}]$, then $\mathsf{ZPEXP} \subseteq \mathtt{i.o.}\mathsf{ESUBEXP}$.

Hardness Results for MCSP. All functions in non-uniform NC^1 reduce to the Minimum Circuit Size Problem via truth-table reductions computable by TC^0 circuits. In particular, if $\mathsf{MCSP} \in \mathsf{TC}^0$ then $\mathsf{NC}^1 = \mathsf{TC}^0$.

1998 ACM Subject Classification F. Theory of Computation

Keywords and phrases Boolean circuits, learning theory, pseudorandomness

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.18

* The first author received support from CNPq grant 200252/2015-1. The second author was supported by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013)/ERC Grant No. 615075. Part of this work was done during a visit of the first author to Oxford supported by the second author's ERC grant.

32nd Computational Complexity Conference (CCC 2017).
Editor: Ryan O'Donnell; Article No. 18; pp. 18:1–18:49

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

1 Introduction

Which classes of functions can be efficiently learned? Answering this question has been a major research direction in computational learning theory since the seminal work of Valiant [68] formalizing efficient learnability.

For concreteness, consider the model of learning with membership queries under the uniform distribution. In this model, the learner is given oracle access to a target Boolean function and aims to produce, with high probability, a hypothesis that approximates the target function well on the uniform distribution. Say that a circuit class $\mathfrak{C}$ is *learnable* in time T if there is a learner running in time T such that for each function $f \in \mathfrak{C}$, when given oracle access to f the learner outputs the description of a Boolean function h approximating f well under the uniform distribution. The hypothesis h is not required to be from the same class $\mathfrak{C}$ of functions. (This and other learning models that appear in our work are defined in Section 2.)

Various positive and conditional negative results are known for natural circuit classes in this model, and here we highlight only a few. Polynomial-time algorithms are known for polynomial-size DNF formulas [34]. Quasi-polynomial time algorithms are known for polynomial-size constant-depth circuits with AND, OR and NOT gates [46] (i.e., AC^0 circuits), and in a recent breakthrough [19], for polynomial-size constant-depth circuits which in addition contain MOD[p] gates, where p is a fixed prime ($\mathsf{AC}^0[p]$ circuits). In terms of hardness, it is known that under certain cryptographic assumptions, the class of polynomial-size constant-depth circuits with threshold gates (TC^0 circuits) is not learnable in sub-exponential time [50]. (We refer to Section 2 for a review of the inclusions between standard circuit classes.)

However, even under strong hardness assumptions, it is still unclear how powerful a circuit class needs to be before learning becomes utterly infeasible. For instance, whether non-trivial learning algorithms exist for classes beyond $\mathsf{AC}^0[p]$ remains a major open problem.

Inspired by [19], we show that a general and surprising *speedup* phenomenon holds unconditionally for learnability of strong enough circuit classes around the border of currently known learning algorithms. Say that a class is *non-trivially learnable* if it is learnable in time $\leq 2^n/n^{w(1)}$, where n is the number of inputs to a circuit in the class, and furthermore the learner is only required to output a hypothesis that is an approximation for the unknown function with inverse polynomial advantage. We show that for "typical" circuit classes such as constant-depth circuits with Mod[m] gates where m is an arbitrary but fixed composite (ACC^0 circuits), constant-depth threshold circuits, formulas and general Boolean circuits, non-trivial learnability in fact implies high-accuracy learnability in time $2^{n^{o(1)}}$, i.e., in sub-exponential time.

▶ **Lemma 1** (Speedup Lemma, Informal Version). *Let $\mathfrak{C}$ be a typical circuit class. Polynomial-size circuits from $\mathfrak{C}$ are non-trivially learnable if and only if polynomial-size circuits from $\mathfrak{C}$ are (strongly) learnable in sub-exponential time. Subexponential-size circuits from $\mathfrak{C}$ are non-trivially learnable if and only if polynomial-size circuits from $\mathfrak{C}$ are (strongly) learnable in quasi-polynomial time.*

Note that the class of *all* Boolean functions is learnable in time $\leq 2^n/n^{\Omega(1)}$ with $\geq 1/n$ advantage simply by querying the function oracle on $2^n/n^{O(1)}$ inputs, and outputting the best constant in $\{0,1\}$ for the remaining (unqueried) positions of the truth-table. Our notion of non-trivial learning corresponds to merely beating this trivial brute-force algorithm – this is sufficient to obtain much more dramatic speedups for learnability of typical circuit classes.

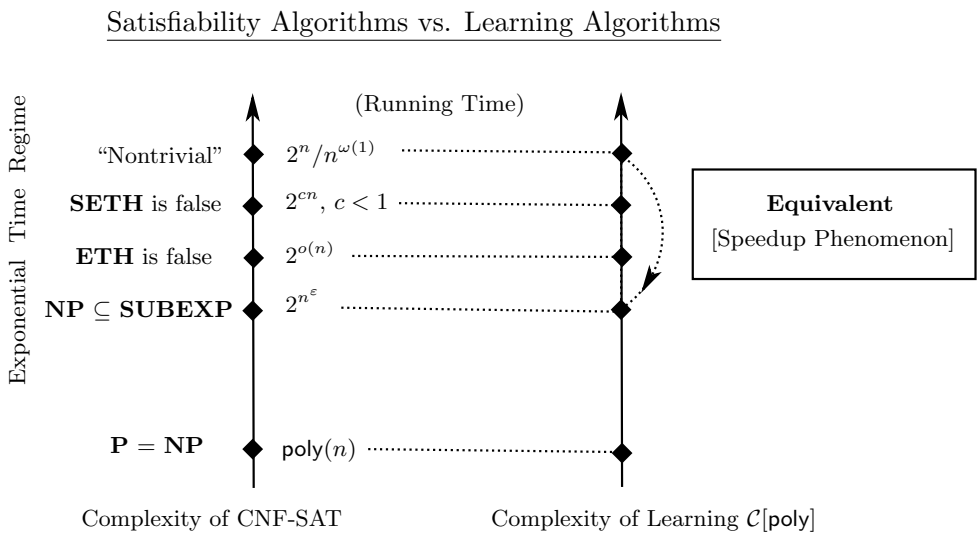


Figure 1 A speedup phenomenon in computational learning theory for typical circuit classes for learning under the uniform distribution with membership queries. The speedup procedure simultaneously boosts accuracy and running time.

In order to provide more intuition for this result, we compare the learning scenario to another widely investigated algorithmic framework. Consider the problem of checking if a circuit from a fixed circuit class is *satisfiable*, a natural generalization of the CNF-SAT problem. Recall that ACC^0 circuits are circuits of constant depth with AND, OR, NOT, and modulo gates. There are non-trivial satisfiability algorithms for ACC^0 circuits of size up to 2^{n^ε}, where $\varepsilon > 0$ depends on the depth and modulo gates [73]. On the other hand, if such circuits admitted a non-trivial learning algorithm, it follows from the Speedup Lemma that polynomial size ACC^0 circuits can be learned in quasi-polynomial time (see Figure 1).

The Speedup Lemma suggests new approaches both to designing learning algorithms and to proving hardness of learning results. To design a quasi-polynomial time learning algorithm for polynomial-size circuits from a typical circuit class, it suffices to obtain a minimal improvement over the trivial brute-force algorithm for sub-exponential size circuits from the same class. Conversely, to conclude that the brute-force learning algorithm is essentially optimal for a typical class of polynomial-size circuits, it suffices to use an assumption under which subexponential-time learning is impossible.

We use the Speedup Lemma to show various structural results about learning. These include equivalences between several previously defined learning models, a dichotomy between sub-exponential time learnability and the existence of pseudo-random function generators in the non-uniform setting, and implications from non-trivial learning to circuit lower bounds.

The techniques we explore have other consequences for complexity theory, such as Karp-Lipton style results for bounded-error exponential time, and results showing hardness of the Minimum Circuit Size Problem for a standard complexity class. In general, our results both exploit and strengthen the rich web of connections between learning, pseudo-randomness and circuit lower bounds, which promises to have further implications for our understanding of these fundamental notions. We now describe these contributions in more detail.

1.1 Summary of Results

We state below informal versions of our main results. We put these results in perspective and compare them to previous work in Section 1.2.

Equivalences for Learning Models. The Speedup Lemma shows that learnability of polynomial size circuits for typical circuit classes is not sensitive to the distinction between randomized sub-exponential time algorithms and randomized non-trivial algorithms. We use the Speedup Lemma to further show that for such classes, learnability for a range of previously defined learning models is equivalent. These include the worst-case and average-case versions of function compression as defined by Chen et al. [20] (see also [64]), and randomized learning with membership and equivalence queries [10].[1] The equivalence between function compression and learning in particular implies that accessing the entire truth table of a function represented by the circuit from the class confers no advantage in principle over having limited access to the truth table.

▶ **Theorem 2** (Equivalences for Learning Models, Informal Version)**.** *The following are equivalent for polynomial-size circuits from a typical circuit class* $\mathfrak{C}$*:*

(1) *Sub-exponential time learning with membership queries.*
(2) *Sub-exponential time learning with membership and equivalence queries.*
(3) *Probabilistic function compression.*
(4) *Average-case probabilistic function compression.*
(5) *Exponential time distinguishability from random functions.*

In particular, in the randomized sub-exponential time regime and when restricted to learning under the uniform distribution, Valiant's model [68] and Angluin's model [10] are equivalent in power with respect to the learnability of typical classes of polynomial size circuits.

A Dichotomy between Learning and Pseudorandomness. It is well-known that if the class of polynomial-size circuits from a class $\mathfrak{C}$ is learnable, then there are no pseudo-random function generators computable in $\mathfrak{C}$, as the learner can be used to distinguish random functions from pseudo-random ones [42]. A natural question is whether the converse is true: can we in general build pseudo-random functions in the class from non-learnability of the class? We are able to use the Speedup Lemma in combination with other techniques to show such a result in the *non-uniform* setting, where the pseudo-random function generator as well as the learning algorithm are non-uniform. As a consequence, for each typical circuit class $\mathfrak{C}$, there is a *dichotomy* between pseudorandomness and learnability – either there are pseudo-random function generators computable in the class, or the class is learnable, but not both.

▶ **Theorem 3** (Dichotomy between Learning and Pseudorandomness, Informal Version)**.** *Let* $\mathfrak{C}$ *be a typical circuit class. There are no pseudo-random function generators computable by polynomial-size circuits from* $\mathfrak{C}$ *that are secure against sub-exponential size Boolean circuits if and only if polynomial-size circuits from* $\mathfrak{C}$ *are learnable non-uniformly in sub-exponential time.*

[1] Our notion of randomized learning with membership and equivalence queries allows the learner's hypothesis to be incorrect on a polynomially small fraction of the inputs.

Nontrivial Learning implies Circuit Lower Bounds. In the algorithmic approach of Williams [70], non-uniform circuit lower bounds against a class $\mathfrak{C}$ of circuits are shown by designing algorithms for satisfiability of $\mathfrak{C}$-circuits that beat the trivial brute-force search algorithm. Williams' approach has already yielded the result that $\mathsf{NEXP} \not\subseteq \mathsf{ACC}^0$ [73].

It is natural to wonder if an analogue of the algorithmic approach holds for learning, and if so, what kinds of lower bounds would follow using such an analogue. We establish such a result – non-trivial learning algorithms yield lower bounds for bounded-error probabilistic exponential time, just as non-trivial satisfiability algorithms yield lower bounds for non-deterministic exponential time. Our connection between learning and lower bounds has a couple of nice features. Our notion of "non-trivial algorithm" can be made even more fine-grained than that of Williams – it is not hard to adapt our techniques to show that it is enough to beat the brute-force algorithm by a super-constant factor for learning algorithms with constant accuracy, as opposed to a polynomial factor in the case of Satisfiability. Moreover, non-trivial learning for bounded-depth circuits yields lower bounds against circuits with the *same* depth, as opposed to the connection for Satisfiability where there is an additive loss in depth [54, 35].

▶ **Theorem 4** (Circuit Lower Bounds from Learning and from Natural Proofs, Informal Version)**.** *Let $\mathfrak{C}$ be any circuit class closed under projections.*

- **(i)** *If polynomial-size circuits from $\mathfrak{C}$ are non-trivially learnable, then (two-sided) bounded-error probabilistic exponential time does not have polynomial-size circuits from $\mathfrak{C}$.*
- **(ii)** *If sub-exponential size circuits from $\mathfrak{C} = \mathsf{ACC}^0$ are non-trivially learnable, then one-sided error probabilistic exponential time does not have polynomial-size circuits from ACC^0.*
- **(iii)** *If there are natural proofs useful against sub-exponential size circuits from $\mathfrak{C}$, then zero-error probabilistic exponential time does not have polynomial-size circuits from $\mathfrak{C}$.*

Observe that the existence of natural proofs against sub-exponential size circuits yields stronger lower bounds than learning and satisfiability algorithms. (We refer to Section 2 for a review of the inclusions between exponential time classes.)

Karp-Lipton Theorems for Probabilistic Exponential Time. Our main results are about learning, but the techniques have consequences for complexity theory. Specifically, our use of pseudo-random generators has implications for the question of Karp-Lipton theorems for probabilistic exponential time. A Karp-Lipton theorem for a complexity class gives a connection between uniformity and non-uniformity, by showing that a non-uniform inclusion of the complexity class also yields a uniform inclusion. Such theorems were known for a range of classes such as NP, PSPACE, EXP, and NEXP [39, 12, 30], but not for bounded-error probabilistic exponential time. We show the first such theorem for bounded-error probabilistic exponential time. A technical caveat is that the inclusion in our consequent is not completely uniform, but requires a logarithmic amount of advice.

▶ **Theorem 5** (Karp-Lipton Theorem for Probabilistic Exponential Time, Informal Version)**.** *If bounded-error probabilistic exponential time has polynomial-size circuits infinitely often, then bounded-error probabilistic exponential time is infinitely often in deterministic exponential time with logarithmic advice.*

Hardness of the Minimum Circuit Size Problem. Our techniques also have consequences for the complexity of the Minimum Circuit Size Problem (MCSP). In MCSP, the input is the truth table of a Boolean function together with a parameter s in unary, and the question

is whether the function has Boolean circuits of size at most s. MCSP is a rare example of a problem in NP which is neither known to be in P or NP-complete. In fact, we don't know much unconditionally about the complexity of this problem. We know that certain natural kinds of reductions cannot establish NP-completeness [49], but until our work, it was unknown whether MCSP is hard for *any* standard complexity class beyond AC^0 [3]. We show the first result of this kind.

▶ **Theorem 6** (Hardness of the Minimum Circuit Size Problem, Informal Version)**.** *The Minimum Circuit Size Problem is hard for polynomial-size formulas under truth-table reductions computable by polynomial-size constant-depth threshold circuits.*

▶ Remark. This work contains several related technical contributions to the research topics mentioned above. We refer to the appropriate sections for more details. Finally, in Section 8 we highlight some open problems and directions that we find particularly attractive.

1.2 Related Work

Speedups in Complexity Theory. We are not aware of any unconditional speedup result of this form involving the time complexity of a natural class of computational problems, under a general computational model. In any case, it is instructive to compare Lemma 1 to a few other speedup theorems in computational complexity.

A classic example is Blum's Speedup Theorem [15]. It implies that there is a recursive function $f\colon \mathbb{N} \to \mathbb{N}$ such that if an algorithm computes this function in time $T(n)$, then there is an algorithm computing f in time $O(\log T(n))$. Lemma 1 differs in an important way. It refers to a natural computational task, while the function provided by Blum's Theorem relies on an artificial construction. Another well-known speedup result is the Linear Speedup Theorem (cf. [55, Section 2.4]). Roughly, it states that if a Turing Machine computes in time $T(n)$, then there is an equivalent Turing Machine that computes in time $T(n)/c$. The proof of this theorem is based on the simple trick of increasing the alphabet size of the machine. It is therefore dependent on the computational model, while Lemma 1 is not.

Perhaps closer to our result are certain conditional derandomization theorems in complexity theory. We mention for concreteness two of them. In [30], it is proved that if $\mathsf{MA} \neq \mathsf{NEXP}$, then $\mathsf{MA} \subseteq \mathtt{i.o.}\mathsf{NTIME}[2^{n^\varepsilon}]/n^\varepsilon$, while in [33], it is shown that if $\mathsf{BPP} \neq \mathsf{EXP}$, then $\mathsf{BPP} \subseteq \mathtt{i.o.pseudo\text{-}}\mathsf{DTIME}[2^{n^\varepsilon}]$. It is possible to interpret these results as computational speedups, but observe that the faster algorithms have either weaker correctness guarantees, or require advice. Lemma 1 on the other hand transforms a non-trivial learning algorithm into a sub-exponential time learning algorithm of the same type.

Further results have been discovered in more restricted computational models. For instance, in the OPP model, [56] proved that if Circuit-SAT has algorithms running in time $2^{(1-\delta)n}$, then it also has OPP algorithms running in time $2^{\varepsilon n}$. In bounded-depth circuit complexity, [8] established among other results that if the Formula Evaluation Problem has uniform TC^0-circuits of size $O(n^k)$, then it also has uniform TC^0-circuits of size $O(n^{1+\varepsilon})$.

If one considers other notions of complexity, we can add to this list several results that provide different, and often rather unexpected, forms of speedup. We mention, for instance, depth reduction in arithmetic circuit complexity (see e.g. [1]), reducing the number of rounds in interactive proofs [13], decreasing the randomness complexity of bounded-space algorithms [52], cryptography in constant locality [11], among many others.

Connections between Pseudorandomness, Learning and Cryptography. There are well-known connections between learning theory, theoretical cryptography and pseudorandomness

(see e.g. [23]). Indeed, pseudorandom distributions lie at the heart of the definition of semantic security [25, 26], which permeates modern cryptography, and to every secure encryption scheme there is a naturally associated hard-to-learn (decryption) problem.

The other direction, i.e., that from a generic hard learning problem it is always possible to construct secure cryptographic schemes and other basic primitives, is much less clear.[2] Following a research line initiated in [31], results more directly related to our work were established in [14]. They proved in particular that private-key encryption and pseudorandom generators exist under a stronger *average-case* hardness-of-learning assumption, where one also considers the existence of a hard distribution over the functions in the circuit class $\mathfrak{C}$.

However, these results and subsequent work leave open the question of whether hardness of learning in the usual case, i.e., the mere assumption that any efficient learner fails on some $f \in \mathfrak{C}$, implies the existence of pseudorandom functions computable by $\mathfrak{C}$-circuits. While there is an extensive literature basing standard cryptographic primitives on a variety of conjecturally hard learning tasks (see e.g., [60] and references therein for such a line of work), to our knowledge Theorem 3 is the first result to establish a *general equivalence* between the existence of pseudorandom functions and the hardness of learning, which holds for *any* typical circuit class. A caveat is that our construction requires non-uniformity, and is established only in the exponential security regime.

Lower Bounds from Learning Algorithms. While several techniques from circuit complexity have found applications in learning theory in the past (see e.g., [46]), Fortnow and Klivans [21] were the first to systematically investigate the connection between learning algorithms and lower bounds in a generic setting.[3]

For *deterministic* learning algorithms using membership and equivalence queries, initial results from [21] and [27] were strengthened and simplified in [44], where it was shown that non-trivial deterministic learning algorithms for $\mathfrak{C}$ imply that $\mathsf{EXP} \not\subseteq \mathfrak{C}$.

The situation for *randomized* algorithms using membership queries is quite different, and only the following comparably weaker results were known. First, [21] proved that randomized polynomial time algorithms imply BPEXP lower bounds. This result was refined in [44], where a certain connection involving sub-exponential time randomized learning algorithms and PSPACE was observed. More recently, [69] combined ideas from [44] and [61] to prove that efficient randomized learning algorithms imply lower bounds for $\mathsf{BPP}/1$, i.e., probabilistic polynomial time with advice. However, in contrast to the deterministic case, obtaining lower bounds from weaker running time assumptions had been elusive.

Indeed, we are not aware of any connection between two-sided non-trivial randomized algorithms and circuit lower bounds, even when considering different algorithmic frameworks in addition to learning. In particular, Theorem 4 (i) seems to be the first result in this direction. It can be seen as an analogue of the connection between satisfiability algorithms and lower bounds established by Williams [70, 73]. But apart from this analogy, the proof of Theorem 4 employs significantly different techniques.

---

[2] Recall that secure private-key encryption is equivalent to the existence of one-way functions, pseudorandom generators and pseudorandom functions, with respect to polynomial time computations (cf. [40]). Nevertheless, not all these equivalences are known to hold when more refined complexity measures are considered, such as circuit depth. In particular, generic constructions of pseudorandom functions from the other primitives are not known in small-depth classes. This can be done under certain *specific* hardness assumptions [50], but here we restrict our focus to generic relations between basic cryptographic primitives.

[3] For a broader survey on connections between algorithms and circuit lower bounds, we refer to [71].

Useful Properties, Natural Properties, and Circuit Lower Bounds. The concept of natural proofs, introduced by Razborov and Rudich [59], has had a significant impact on research on unconditional lower bounds. Recall that a property $\mathcal{P}$ of Boolean functions is a natural property against a circuit class $\mathfrak{C}$ if it is: (1) efficiently computable (constructivity); (2) rejects all $\mathfrak{C}$-functions, and accepts at least one "hard" function (usefulness), and (3) is satisfied by most Boolean functions (denseness). In case $\mathcal{P}$ satisfies only conditions (1) and (2), is it said to be useful against $\mathfrak{C}$.

There are natural properties against $\mathsf{AC}^0[p]$ circuits, when p is prime [59]. But under standard cryptographic assumptions, there is no natural property against TC^0 [50]. Consequently, the situation for classes contained in $\mathsf{AC}^0[p]$ and for those that contain TC^0 is reasonably well-understood. More recently, [74] (see also [30]) proved that if $\mathsf{NEXP} \not\subseteq \mathfrak{C}$ then there are useful properties against $\mathfrak{C}$. This theorem combined with the lower bound from [73] show that ACC^0 admits useful properties.

Given these results, the existence of natural properties against ACC^0 has become one of the most intriguing problems in connection with the theory of natural proofs. Theorem 4 (*iii*) shows that if there are P-natural properties against sub-exponential size ACC^0 circuits, then $\mathsf{ZPEXP} \not\subseteq \mathsf{ACC}^0$. This would lead to an improvement of Williams' celebrated lower bound which does not seem to be accessible using his techniques alone.[4]

Karp-Lipton Theorems in Complexity Theory. Karp-Lipton theorems are well-known results in complexity theory relating non-uniform circuit complexity and uniform collapses. A theorem of this form was first established in [39], where they proved that if $\mathsf{NP} \subseteq \mathsf{Circuit}[\mathsf{poly}]$, then the polynomial time hierarchy collapses. This result shows that non-uniform circuit lower bounds cannot be avoided if our goal is a complete understanding of uniform complexity theory.

Since their fundamental work, many results of this form have been discovered for complexity classes beyond NP. In some cases, the proof required substantially new ideas, and the new Karp-Lipton collapse led to other important advances in complexity theory. Below we discuss the situation for two exponential complexity classes around BPEXP, which is connected to Theorem 5.

A stronger Karp-Lipton theorem for EXP was established in [12], using techniques from interactive proofs and arithmetization. An important application of this result appears in [17] in the proof that $\mathsf{MAEXP} \not\subseteq \mathsf{Circuit}[\mathsf{poly}]$. This is still one of the strongest known non-uniform lower bounds. For NEXP, a Karp-Lipton collapse was proved in [30]. This time the proof employed the easy witness method and techniques from pseudorandomness, and the result plays a fundamental role in Williams' framework [70], which culminated in the proof that $\mathsf{NEXP} \not\subseteq \mathsf{ACC}^0$ [73]. (We mention that a Karp-Lipton theorem for $\mathsf{EXP}^{\mathsf{NP}}$ has also been established in [18].) Karp-Lipton collapse theorems are known for a few other complexity classes contained in EXP, and they have found applications in a variety of contexts in algorithms and complexity theory (see e.g., [75, 22]).

Despite this progress on proving Karp-Lipton collapses for exponential time classes, there is no published work on such for probabilistic classes. Theorem 5 is the first such result for the class BPEXP.

[4] The result that P-natural properties against sub-exponential size circuits yield ZPEXP lower bounds was also obtained in independent work by Russell Impagliazzo, Valentine Kabanets and Ilya Volkovich (private communication).

The Minimum Circuit Size Problem. The Minimum Circuit Size Problem (MCSP) and its variants has received a lot of attention in both applied and theoretical research. Its relevance in practice is clear. From a theoretical point of view, it is one of the few natural problems in NP that has not been shown to be in P or NP-complete. The hardness of MCSP is also connected to certain fundamental problems in proof complexity (cf. [45, 58]).

Interestingly, a well-understood variant of MCSP is the Minimum DNF Size Problem, for which both NP-hardness [48] and near-optimal hardness of approximation have been established [6, 43]. However, despite the extensive literature on the complexity of the MCSP problem [38, 3, 4, 29, 7, 49, 29, 5, 28], and the intuition that it must also be computationally hard, there are few results providing evidence of its difficulty. Among these, we highlight the unconditional proof that $\mathsf{MCSP} \notin \mathsf{AC}^0$ [3], and the reductions showing that $\mathsf{Factoring} \in \mathsf{ZPP}^{\mathsf{MCSP}}$ [3] and $\mathsf{SZK} \subseteq \mathsf{BPP}^{\mathsf{MCSP}}$ [4]. The lack of further progress has led to the formulation and investigation of a few related problems, for which some additional results have been obtained (cf. [3, 7, 5, 28]).

More recently, [49] provided some additional explanation for the difficulty of proving hardness of MCSP. They unconditionally established that a class of local reductions that have been used for many other NP-completeness proofs cannot work, and that the existence of a few other types of reductions would have significant consequences in complexity theory. Further results along this line appear in [29].

Theorem 6 contributes to our understanding of the difficulty of MCSP by providing the first hardness results for a standard complexity class beyond AC^0. We hope this result will lead to further progress on the quest to determine the complexity of this elusive problem.[5]

1.3 Main Techniques

1.3.1 Overview

Our results are obtaining via a mixture of techniques from learning theory, computational complexity, pseudo-randomness and circuit complexity. We refer to Figure 2 for a web of connections involving the theorems stated in Section 1.1 and the methods employed in the proofs. We start with an informal description of most of the techniques depicted in Figure 2, with pointers to some relevant references.[6]

Nisan-Wigderson Generator [51]. The NW-Generator allows us to convert a function $f\colon \{0,1\}^n \to \{0,1\}$ into a family of functions $\mathsf{NW}(f)$. Crucially, if an algorithm A is able to distinguish $\mathsf{NW}(f)$ from a random function, there is a reduction that only needs oracle access to f and A, and that can be used to weakly approximate f. The use of the NW-Generator in the context of learning, for a function f that is not necessarily hard, appeared recently in [19].[7]

[5] We have learned from Eric Allender (private communication) that in independent work with Shuichi Hirahara, they have shown some hardness results for the closely related problem of whether a string has high KT complexity. These results do not yet seem to transfer to MCSP and its variants. In addition, we have learned from Valentine Kabanets (private communication) that in recent independent work with Russell Impagliazzo and Ilya Volkovich, they have also obtained some results on the computational hardness of MCSP.

[6] This is not a comprehensive survey of the original use or appearance of each method. It is included here only as a quick guide to help the reader to assimilate the main ideas employed in the proofs.

[7] Interestingly, another unexpected and somewhat related use of the NW-generator appears in proof complexity (see e.g., [57] and references therein).

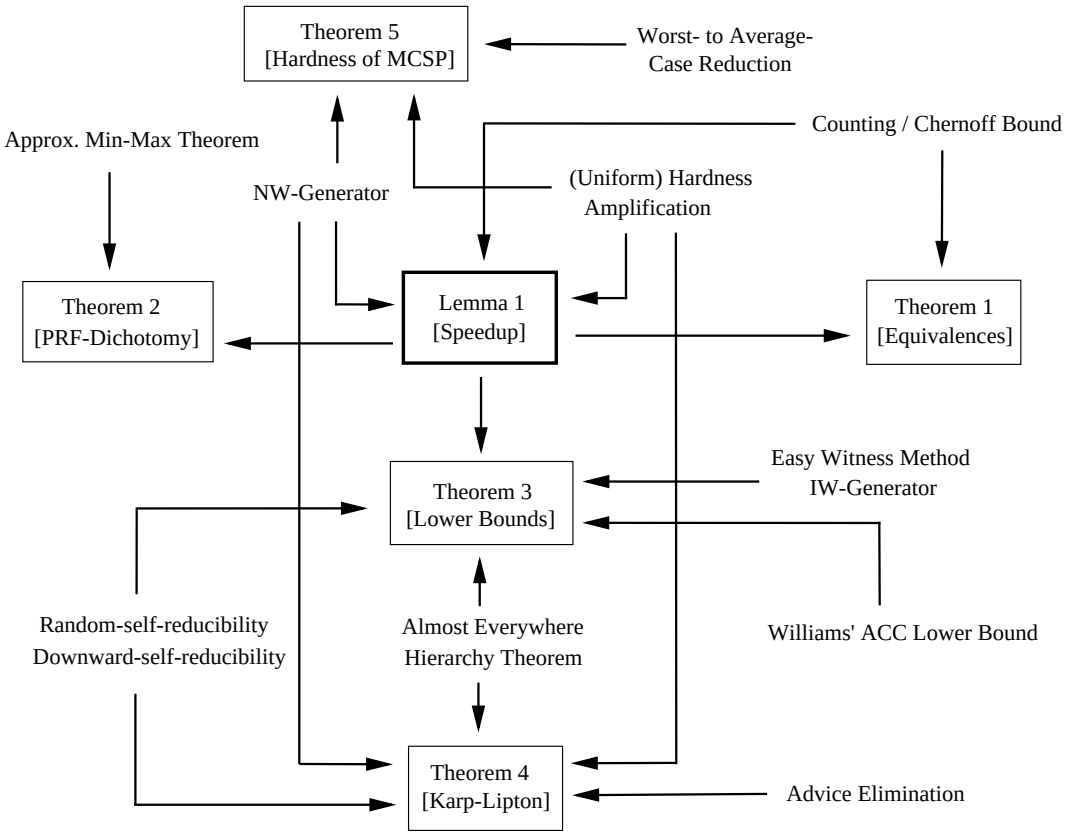


Figure 2 An overview of the main techniques employed in the proof of each result discussed in Section 1.1. An arrow from P to Q indicates that the proof of Q relies on P.

(Uniform) Hardness Amplification. This is a well-known technique in circuit complexity (cf. [24]), allowing one to produce a not much more complex function $\tilde{g}\colon \{0,1\}^{m(n)} \to \{0,1\}$, given oracle access to some function $g\colon \{0,1\}^n \to \{0,1\}$, that is much harder to approximate than g. The uniform formulation of this result shows that a weak approximator for $\tilde{g}$ can be converted into a strong approximator for g. The connection to learning was explicitly observed in [16].

Counting and Concentration Bounds. This is a standard argument which allows one to prove that most Boolean functions on n-bit inputs cannot be approximated by Boolean circuits of size $\leq 2^n/n^{\omega(1)}$ (Lemma 21). In particular, learning algorithm running in non-trivial time can only successfully learn a negligible fraction of all Boolean functions.

Small-Support Min-Max Theorem [9, 47]. This is an approximate version of the well-known min-max theorem from game theory. It provides a bound on the support size of the mixed strategies. To prove Theorem 3, we consider a game between a function family generator and a candidate distinguisher, and this result allows us to move from a family of distinguishers against different classes of functions to a single universal distinguisher of bounded complexity.

Worst-Case to Average-Case Reduction. The NW-Generator and hardness amplification can be used to boost a very weak approximation into a strong one. In some circuit classes such as NC^1, a further reduction allows one to obtain a circuit that is correct on every input with high probability (see e.g. [2]). This is particularly useful when proving hardness results for MCSP.

Easy Witness Method [37] and Impagliazzo-Wigderson Generator [32]. The easy witness method is usually employed as a win-win argument: either a verifier accepts a string encoded by a small circuit, or every accepted string has high worst-case circuit complexity. No matter the case, it can be used to our advantage, thanks to the generator from [32] that transforms a worst-case hard string (viewed as a truth table) into a pseudorandom distribution of strings.

(Almost Everywhere) Hierarchy Theorems. A difficulty when proving Theorems 4 and 5 is that there are no known tight hierarchy theorems for randomized time. Our approach is therefore indirect, relying on the folklore result that bounded-space algorithms can diagonalize on every input length against all bounded-size circuits (Lemma 38 and Corollary 39).

Random-self-reducibility and Downward-self-reducibility. These are important notions of self-reducibility shared by certain functions. Together, they can be used via a recursive procedure to obtain from a learning algorithm for such a function, which requires oracle access to the function, a standard randomized algorithm computing the same function [33, 65].

Advice Elimination. This idea is important in the contrapositive argument establishing Theorem 5. Assuming that a certain deterministic simulation of a function in BPEXP is not successful, it is not clear how to determine on each input length a "bad" string of that length for which the simulation fails. Such bad strings are passed as advice in our reduction, and in order to eliminate the dependency on them, we use an advice-elimination strategy from [65].

1.3.2 Sketch of Proofs

We describe next in a bit more detail how the techniques described above are employed in the proof of our main results. We stress that the feasibility of all these arguments crucially depend on the parameters associated to each result and technique. However, for simplicity our focus here will be on the qualitative connections.

Lemma 1 (Speedup Lemma). Given query access to a function $f \in \mathfrak{C}$ that we would like to learn to high accuracy, the first idea is to notice that if there is a distinguisher against $\mathsf{NW}(f)$, then we can non-trivially approximate f using membership queries. But since this is not the final goal of a strong learning algorithm, we consider $\mathsf{NW}(\tilde{f})$, the generator applied to the amplified version of f. Using properties of the NW-generator and hardness amplification, it follows that if there is a distinguisher against $\mathsf{NW}(\tilde{f})$, it is possible to approximate $\tilde{f}$, which in turn provides a strong approximator for f. (A similar strategy is employed in [19], where a natural property is used instead of a distinguisher.)

Next we use the assumption that $\mathfrak{C}$ has non-trivial learning algorithms to obtain a distinguisher against $\mathfrak{C}$. (For this approach to work, it is fundamental that the functions in $\mathsf{NW}(\tilde{f}) \subseteq \mathfrak{C}$. In other words, the reductions discussed above should not blow-up the complexity of the involved functions by too much. For this reason, $\mathfrak{C}$ must be a sufficiently strong circuit class.) By a counting argument and a concentration bound, while a non-trivial

learning algorithm will weakly learn every function in $\mathfrak{C}$, it must fail to learn a random Boolean function with high probability. We apply this idea to prove that a non-trivial learner can be used as a distinguisher against $\mathsf{NW}(\tilde{f})$.[8]

These techniques can therefore be combined in order to boost a non-trivial learner for $\mathfrak{C}$ into a high-accuracy learner for $\mathfrak{C}$. This takes care of the accuracy amplification. The running time speedup comes from the efficiency of the reductions involved, and from the crucial fact that each function in $\mathsf{NW}(\tilde{f})$ is a function over $m \ll n$ input bits. In particular, the non-trivial but still exponential time learning algorithm for $\mathfrak{C}$ only needs to be invoked on Boolean functions over m input bits. (This argument only sketches one direction in Lemma 1.)

Theorem 2 (Learning Equivalences). At the core of the equivalence between all learning and compression models in Theorem 2 is the idea that in each case we can obtain a certain distinguisher from the corresponding algorithm. Again, this makes fundamental use of counting and concentration bounds to show that non-trivial algorithms can be used as distinguishers. On the other hand, the Speedup Lemma shows that a distinguisher can be turned into a sub-exponential time randomized learning algorithm that requires membership queries only.

In some models considered in the equivalence, additional work is necessary. For instance, in the learning model where equivalence queries are allowed, they must be simulated by the distinguisher. For exact compression, a hypothesis output by the sub-exponential time learner might still contain errors, and these need to be corrected by the compression algorithm. A careful investigation of the parameters involved in the proof make sure the equivalences indeed hold.

Theorem 3 (Dichotomy between Learning and PRFs). It is well-known that the existence of learning algorithms for a class $\mathfrak{C}$ implies that $\mathfrak{C}$-circuits cannot compute pseudorandom functions. Using the Speedup Lemma, it follows that the existence of *non-trivial* learning algorithms for $\mathfrak{C}$ implies that $\mathfrak{C}$ cannot compute *exponentially* secure pseudorandom functions.

For the other direction, assume that every samplable family $\mathcal{F}$ of functions from $\mathfrak{C}$ can be distinguished from a random function by some procedure $D_{\mathcal{F}}$ of sub-exponential complexity. By introducing a certain two-player game (Section 4.1), we are able to employ the small-support min-max theorem to conclude that there is a single circuit of bounded size that distinguishes every family of functions in $\mathfrak{C}$ from a random function. In turn, the techniques behind the Speedup Lemma imply that every function in $\mathfrak{C}$ can be learned in sub-exponential time.

We remark that the non-uniformity in the statement of Theorem 3 comes from the application of a non-constructive min-max theorem.

Theorem 4 (Lower Bounds from Non-trivial Learning and Natural Proofs). Here we combine the Speedup Lemma with the self-reducibility approach from [33, 65, 21, 44] and other standard arguments. Assuming a non-trivial learning algorithm for $\mathfrak{C}$, we first boost it to a high-accuracy *sub-exponential* time learner. Now if $\mathsf{PSPACE} \not\subseteq \mathfrak{C}$ we are done, since

[8] A natural question is whether a non-trivial learning directly implies the existence of a BPP-natural property, which would mean the speedup follows from the main result of [19] in a black-box way. However, this does not appear to be the case – the learner might learn successfully with probability strictly between 1/3 and 2/3 for some truth tables, which would violate the BPP-promise on constructivity of the putative natural property corresponding to the distinguisher.

$\mathsf{PSPACE} \subseteq \mathsf{BPEXP}$. Otherwise, using a special *self-reducible* complete function $f \in \mathsf{PSPACE}$ [65], we are able obtain from a sub-exponential time *learning* algorithm for f a *sub-exponential* time *decision* algorithm computing f. Using the completeness of f and a strong hierarchy theorem for bounded-space algorithms, standard techniques allow us to translate the hardness of PSPACE against bounded-size circuits and the non-trivial upper bound on the randomized complexity of f into a non-uniform circuit lower bound for randomized exponential time. A win-win argument is used crucially to establish that no depth blow-up is necessary when moving from a non-trivial algorithm for (depth-d)-$\mathfrak{C}$ to a (depth-d)-$\mathfrak{C}$ circuit lower bound. For $\mathfrak{C} = \mathsf{ACC}^0$, we combine certain complexity collapses inside the argument with Williams' lower bound [73].

In order to obtain even stronger lower bounds from natural properties against sub-exponential size circuits, we further combine this approach with an application of the easy witness method. This and other standard techniques lead to the collapse $\mathsf{BPEXP} = \mathsf{ZPEXP}$, which strengthens the final circuit lower bound.

Theorem 5 (Karp-Lipton Collapse for Probabilistic Time). This result does not rely on the Speedup Lemma, but its argument is somewhat more technically involved than the proof of Theorem 4. The result is established in the contrapositive. Assuming that an attempted derandomization of BPEXP fails, we show that polynomial space can be simulated in sub-exponential randomized time. Arguing similarly to the proof of Theorem 4, we conclude that there are functions in randomized exponential time that are not infinitely often computed by small circuits.

The first difficulty is that the candidate derandomization procedure on n-bit inputs requires the use of the NW-generator applied to a function on n^c-bit inputs, due to our setting of parameters. However, in order to invoke the self-reducibility machinery, we need to make sure the generator can be broken on *every* input length, and not on infinitely many input lengths. To address this, we introduce logarithmic advice during the simulation, indicating which input length in $[n^c, (n+1)^c]$ should be used in the generator. This amount of advice is reflected in the statement of the theorem.

A second difficulty is that if the derandomization fails on some input string of length n, it is important in the reduction to know a "bad" string with this property. For each input length, a bad string is passed as advice to the learning-to-decision reduction (this is the second use of advice in the proof). This time we are able to remove the advice using an advice-elimination technique, which makes use of self-correctability as in [65]. Crucially, the advice elimination implies that randomized exponential time *without advice* is not infinitely often contained in $\mathfrak{C}$, which completes the proof of the contrapositive of Theorem 5.

Theorem 6 (Hardness of MCSP). Recall that this result states that MCSP is hard for NC^1 with respect to non-uniform TC^0 reductions. The proof of Theorem 6 explores the fine-grained complexity of the Nisan-Wigderson reconstruction procedure and of the hardness amplification reconstruction algorithm. In order words, the argument depends on the combined circuit complexity of the algorithm that turns a distinguisher for $\mathsf{NW}(\tilde{f})$ into a high-accuracy approximating circuit for f, under the notation of the proof sketch for Lemma 1. This time we obtain a distinguisher using an oracle to MCSP. It is possible to show that this reduction can be implemented in non-uniform TC^0.

Observe that the argument just sketched only provides a randomized reduction that approximates the initial Boolean function f under the uniform distribution. But Theorem 6 requires a *worst-case* reduction from NC^1 to MCSP. In other words, we must be able to

compute any NC^1 function correctly on every input. This can be achieved using that there are functions in NC^1 that are NC^1-hard under TC^0-reductions, and that in addition admit randomized worst-case to average-case reductions computable in TC^0. Using non-uniformity, randomness can be eliminated by a standard argument. Altogether, this completes the proof that NC^1 reduces to MCSP via a non-uniform TC^0 computation.

These proofs provide a few additional examples of the use of pseudorandomness in contexts where this notion is not intrinsic to the result under consideration. For instance, the connection between non-trivial learning algorithms and lower bounds (Theorem 4), the Karp-Lipton collapse for probabilistic exponential time (Theorem 5), and the hardness of the Minimum Circuit Size Problem (Theorem 6) are statements that do not explicitly refer to pseudorandomness. Nevertheless, the arguments discussed above rely on this concept in fundamental ways. This motivates a further investigation of the role of pseudorandomness in complexity theory, both in terms of finding more applications of the "pseudorandom method", as well as in discovering alternative proofs relying on different techniques.

2 Preliminaries and Notation

2.1 Boolean Function Complexity

We use $\mathcal{F}_m$ to denote the set of all Boolean functions $f\colon \{0,1\}^m \to \{0,1\}$. If W is a probability distribution, we use $w \sim W$ to denote an element sampled according to W. Similarly, for a finite set A, we use $a \sim A$ to denote that a is selected uniformly at random from A. Under this notation, $f \in \mathcal{F}_m$ represents a fixed function, while $f \sim \mathcal{F}_m$ is a uniformly random function. For convenience, we let $\mathcal{U}_n \stackrel{\text{def}}{=} \{0,1\}^n$. Following standard notation, $X \equiv Y$ denotes that random variables X and Y have the same distribution. We use standard asymptotic notation such as $o(\cdot)$ and $O(\cdot)$, and it always refer to a parameter $n \to \infty$, unless stated otherwise.

We say that $f, g \in \mathcal{F}_n$ are ε-close if $\Pr_{x\sim\mathcal{U}_n}[f(x) = g(x)] \geq 1 - \varepsilon$. We say that $h \in \mathcal{F}_n$ computes f with advantage δ if $\Pr_{x\sim\mathcal{U}_n}[f(x) = h(x)] \geq 1/2 + \delta$. It will sometimes be convenient to view a Boolean function $f \in \mathcal{F}_m$ as a subset of $\{0,1\}^m$ in the natural way.

We often represent Boolean functions as strings via the truth table mapping. Given a Boolean function $f \in \mathcal{F}_n$, $\mathtt{tt}(f)$ is the 2^n-bit string which represents the truth table of f in the standard way, and conversely, given a string $y \in \{0,1\}^{2^n}$, $\mathtt{fn}(y)$ is the Boolean function in $\mathcal{F}_n$ whose truth table is represented by y.

Let $\mathfrak{C} = \{\mathcal{C}_n\}_{n\in\mathbb{N}}$ be a class of Boolean functions, where each $\mathcal{C}_n \subseteq \mathcal{F}_n$. Given a language $L \subseteq \{0,1\}^*$, we write $L \in \mathfrak{C}$ if for every large enough n we have that $L_n \stackrel{\text{def}}{=} \{0,1\}^n \cap L$ is in $\mathcal{C}_n$. Often we will abuse notation and view $\mathfrak{C}$ as a class of Boolean circuits. For convenience, we use number of wires to measure circuit size. We denote by $\mathfrak{C}[s(n)]$ the set of n-variable $\mathfrak{C}$-circuits of size at most $s(n)$. As usual, we say that a uniform complexity class Γ is contained in $\mathfrak{C}[\mathsf{poly}(n)]$ if for every $L \in \Gamma$ there exists $k \geq 1$ such that $L \in \mathfrak{C}[n^k]$.

We say that $\mathfrak{C}$ is *typical* if $\mathfrak{C} \in \{\mathsf{AC}^0, \mathsf{AC}^0[p], \mathsf{ACC}^0, \mathsf{TC}^0, \mathsf{NC}^1, \mathsf{Formula}, \mathsf{Circuit}\}$. Recall that

$$\mathsf{CNF}, \mathsf{DNF} \subsetneq \mathsf{AC}^0 \subsetneq \mathsf{AC}^0[p] \subsetneq \mathsf{ACC}^0 \subseteq \mathsf{TC}^0 \subseteq \mathsf{NC}^1 = \mathsf{Formula}[\mathsf{poly}] \subseteq \mathsf{Circuit}[\mathsf{poly}].$$

We assume for convenience that TC^0 is defined using (unweighted) majority gates instead of weighted threshold gates. Also, while NC^1 typically refers to circuits of polynomial size and logarithmic depth, we consider the generalized version where $\mathsf{NC}^1[s]$ is the class of languages computed by circuits of size $\leq s$ and depth $\leq \log s$.

While we restrict our statements to typical classes, it is easy to see that they generalize to most circuit classes of interest. When appropriate we use $\mathfrak{C}_d$ to restrict attention to $\mathfrak{C}$-circuits of depth at most d. In this work, we often find it convenient to suppress the dependence on d, which is implicit for instance in the definition of a circuit family from a typical bounded-depth circuit class, such as the first four typical classes in the list above. It will be clear from the context whether the quantification over d is existential or universal.

Given a sequence of Boolean functions $\{f_n\}_{n\in\mathbb{N}}$ with $f_n\colon \{0,1\}^n \to \{0,1\}$, we let $\mathfrak{C}^f$ denote the extension of $\mathfrak{C}$ that allows $\mathcal{C}_n$-circuits to have oracle gates computing f_n.

For a complexity class Γ and a language $L \subseteq \{0,1\}^*$, we say that $L \in \mathtt{i.o}.\Gamma$ if there is a language $L' \in \Gamma$ such that $L_n = L'_n$ for infinitely many values of n. Consequently, if $\Gamma_1 \nsubseteq \mathtt{i.o}.\Gamma_2$ then there is a language in Γ_1 that disagrees with each language in Γ_2 on every large enough input length.

Recall the following diagram of class inclusions involving standard complexity classes:[9]

$$\mathsf{ZPP} \subseteq \mathsf{RP} \begin{array}{c} \subsetneq \\ \subseteq \end{array} \begin{array}{c} \mathsf{NP} \\ \mathsf{BPP} \end{array} \begin{array}{c} \subseteq \\ \subsetneq \end{array} \mathsf{MA} \subseteq \mathsf{PSPACE} \subseteq \mathsf{EXP} \subseteq \mathsf{ZPEXP} \subseteq \mathsf{REXP} \begin{array}{c} \subsetneq \\ \subseteq \end{array} \begin{array}{c} \mathsf{NEXP} \\ \mathsf{BPEXP} \end{array} \begin{array}{c} \subseteq \\ \subsetneq \end{array} \mathsf{MAEXP} \subseteq \mathsf{EXPSPACE}.$$

In order to avoid confusion, we fix the following notation for exponential complexity classes. E refers to languages computed in time $2^{O(n)}$. EXP refers to languages computed with bounds of the form 2^{n^c} for some $c \in \mathbb{N}$. SUBEXP denotes complexity 2^{n^ε} for a fixed but arbitrarily small $\varepsilon > 0$. Finally, $\mathsf{ESUBEXP}$ refers to a bound of the form $2^{2^{n^\varepsilon}}$, again for a fixed but arbitrarily small $\varepsilon > 0$. These conventions are also used for the $\mathsf{DSPACE}(\cdot)$ and $\mathsf{BPTIME}(\cdot)$ variants, such as BPE, $\mathsf{BPSUBEXP}$ and $\mathsf{EXPSPACE}$. For instance, a language $L \subseteq \{0,1\}^*$ is in $\mathsf{BPSUBEXP}$ if for every $\varepsilon > 0$ there is a bounded-error randomized algorithm that correctly computes L in time $\leq 2^{n^\varepsilon}$ on every input of length n, provided that n is sufficiently large. For quasi-polynomial time classes such as RQP and BPQP, the convention is that for each language in the class there is a constant $c \geq 1$ such that the corresponding algorithm runs in time at most $O(n^{(\log n)^c})$.

We will use a few other standard notions, and we refer to standard textbooks in computational complexity and circuit complexity for more details.

2.2 Learning and Compression Algorithms

The main learning model with which we concern ourselves is PAC learning under the uniform distribution with membership queries.

▶ **Definition 7** (Learning Algorithms)**.** Let $\mathfrak{C}$ be a circuit class. Given a size function $s\colon \mathbb{N} \to \mathbb{N}$ and a time function $T\colon \mathbb{N} \to \mathbb{N}$, we say that $\mathfrak{C}[s]$ has $(\varepsilon(n), \delta(n))$-learners running in time $T(n)$ if there is a randomized oracle algorithm A^f (the learner) such that for every large enough $n \in \mathbb{N}$:

- For every function $f \in \mathfrak{C}[s(n)]$, given oracle access to f, with probability at least $1 - \delta(n)$ over its internal randomness, $A^f(1^n)$ outputs a Boolean circuit h such that $\Pr_{x\sim\mathcal{U}_n}[f(x) \neq h(x)] \leq \varepsilon(n)$.
- For every function f, $A^f(1^n)$ runs in time at most $T(n)$.

[9] Non-uniform lower bounds against unrestricted polynomial size circuits are currently known only for MAEXP, the exponential time analogue of MA [17].

It is well-known that the confidence of a learning algorithm can be amplified without significantly affecting the running time (cf. [41]), and unless stated otherwise we assume that $\delta(n) = 1/n$.

A *weak learner* for $\mathfrak{C}[s(n)]$ is a $(1/2 - 1/n^c, 1/n)$-learner, for some fixed $c > 0$ and sufficiently large n. We say $\mathfrak{C}[s]$ has *strong learners* running in time T if for each $k \geq 1$ there is a $(1/n^k, 1/n)$-learner for $\mathfrak{C}[s]$ running in time T. Different values for the accuracy parameter k can lead to different running times, but we will often need only a fixed large enough k when invoking the learning algorithm. On the other hand, when proving that a class has a strong learner, we show that the claimed asymptotic running time holds for all fixed $k \in \mathbb{N}$. For simplicity, we may therefore omit the dependence of T on k. We say that $\mathfrak{C}[s]$ has *non-trivial learners* if it has $(1/2 - 1/n^k, 1/n)$-learners running in time $T(n) = 2^n/n^{\omega(1)}$, for some fixed $k \in \mathbb{N}$.

We also discuss randomized learning under the uniform distribution with membership queries and *equivalence queries* [10]. In this stronger model, the learning algorithm is also allowed to make queries of the following form: Is the unknown function f computed by the Boolean circuit C? Here C is an efficient representation of a Boolean circuit produced be the learner. The oracle answers "yes" if the Boolean function computed by C is f; otherwise it returns an input x such that $C(x) \neq f(x)$.

▶ **Definition 8** (Compression Algorithms)**.** Given a circuit class $\mathfrak{C}$ and a size function $s\colon \mathbb{N} \to \mathbb{N}$, a compression algorithm for $\mathfrak{C}[s]$ is an algorithm A for which the following hold:

- Given an input $y \in \{0,1\}^{2^n}$, A outputs a circuit D (not necessarily in $\mathfrak{C}$) of size $o(2^n/n)$ such that if $\mathtt{fn}(y) \in \mathfrak{C}[s(n)]$ then D computes $\mathtt{fn}(y)$.
- A runs in time polynomial in $|y| = 2^n$.

We say $\mathfrak{C}[s]$ admits compression if there is a (polynomial time) compression algorithm for $\mathfrak{C}[s]$.

We will also consider the following variations of compression. If the algorithm is probabilistic, producing a correct circuit with probability $\geq 2/3$, we say $\mathfrak{C}[s]$ has *probabilistic compression*. If the algorithm produces a circuit D which errs on at most $\varepsilon(n)$ fraction of inputs for $\mathtt{fn}(y)$ in $\mathfrak{C}[s]$, we say that A is an *average-case compression algorithm with error* $\varepsilon(n)$. We define correspondingly what it means for a circuit class to have average-case compression or probabilistic average-case compression.

2.3 Natural Proofs and the Minimum Circuit Size Problem

We say that $\mathfrak{R} = \{\mathcal{R}_n\}_{n\in\mathbb{N}}$ is a *combinatorial property* (of Boolean functions) if $\mathcal{R}_n \subseteq \mathcal{F}_n$ for all n. We use $L_{\mathfrak{R}}$ to denote the language of truth-tables of functions in $\mathfrak{R}$. Formally, $L_{\mathfrak{R}} = \{y \mid y = \mathtt{tt}(f) \text{ for some } f \in \mathcal{R}_n \text{ and } n \in \mathbb{N}\}$.

▶ **Definition 9** (Natural Properties [59])**.** Let $\mathfrak{R} = \{\mathcal{R}_n\}$ be a combinatorial property, $\mathfrak{C}$ a circuit class, and $\mathfrak{D}$ a (uniform or non-uniform) complexity class. We say that $\mathfrak{R}$ is a $\mathfrak{D}$-natural property useful against $\mathfrak{C}[s(n)]$ if there is $n_0 \in \mathbb{N}$ such that the following holds:

(i) *Constructivity.* $L_{\mathfrak{R}} \in \mathfrak{D}$.
(ii) *Density.* For every $n \geq n_0$, $\Pr_{f\sim\mathcal{F}_n}[f \in \mathcal{R}_n] \geq 1/2$.
(iii) *Usefulness.* For every $n \geq n_0$, we have $\mathcal{R}_n \cap \mathcal{C}_n[s(n)] = \emptyset$.

▶ **Definition 10** (Minimum Circuit Size Problem)**.** Let $\mathfrak{C}$ be a circuit class. The Minimum Circuit Size Problem for $\mathfrak{C}$, abbreviated as MCSP-$\mathfrak{C}$, is defined as follows:

- *Input.* A pair (y, s), where $y \in \{0,1\}^{2^n}$ for some $n \in \mathbb{N}$, and $1 \leq s \leq 2^n$ is an integer (inputs not of this form are rejected).
- *Question.* Does $\mathtt{fn}(y)$ have $\mathfrak{C}$-circuits of size at most s?

We also define a variant of this problem, where the circuit size is not part of the input.

▶ **Definition 11** (Unparameterized Minimum Circuit Size Problem)**.** Let $\mathfrak{C}$ be a circuit class, and $s\colon \mathbb{N} \to \mathbb{N}$ be a function. The Minimimum Circuit Size Problem for $\mathfrak{C}$ with parameter s, abbreviated as MCSP-$\mathfrak{C}[s]$, is defined as follows:

- *Input.* A string $y \in \{0,1\}^{2^n}$, where $n \in \mathbb{N}$ (inputs not of this form are rejected).
- *Question.* Does $\mathtt{fn}(y)$ have $\mathfrak{C}$-circuits of size at most $s(n)$?

Note that a dense property useful against $\mathfrak{C}[s(n)]$ is a dense subset of the complement of MCSP-$\mathfrak{C}[s]$.

2.4 Randomness and Pseudorandomness

▶ **Definition 12** (Pseudorandom Generators)**.** Let $\ell\colon \mathbb{N} \to \mathbb{N}$, $h\colon \mathbb{N} \to \mathbb{N}$ and $\varepsilon\colon \mathbb{N} \to [0,1]$ be functions, and let $\mathfrak{C}$ be a circuit class. A sequence $\{G_n\}$ of functions $G_n\colon \{0,1\}^{\ell(n)} \to \{0,1\}^n$ is an (ℓ, ε) pseudorandom generator (PRG) against $\mathfrak{C}[h(n)]$ if for any sequence of circuits $\{D_n\}$ with $D_n \in \mathfrak{C}[h(n)]$ and for all large enough n,

$$\left| \Pr_{w \sim U_n}[D_n(w) = 1] - \Pr_{x \sim \mathcal{U}_{\ell(n)}}[D_n(G_n(x)) = 1] \right| \leq \varepsilon(n).$$

The pseudorandom generator is called quick if its range is computable in time $2^{O(\ell(n))}$.

▶ **Theorem 13** (PRGs from computational hardness [51, 32])**.** *Let $s\colon \mathbb{N} \to \mathbb{N}$ be a time-constructible function such that $n \leq s(n) \leq 2^n$ for every $n \in \mathbb{N}$. There is a constant $c > 0$ and an algorithm which, given as input n in unary and the truth table of a Boolean function on $s^{-1}(n)$ bits which does not have circuits of size n^c, computes the range of a $(\ell(n), 1/n)$ pseudorandom generator against* Circuit$[n]$ *in time $2^{O(\ell(n))}$, where $\ell(n) = O((s^{-1}(n))^2/\log n)$.*

▶ **Definition 14** (Distinguishers and Distinguishing Circuits)**.** Given a probability distribution W_n with $\mathsf{Support}(W_n) \subseteq \{0,1\}^n$ and a Boolean function $h_n\colon \{0,1\}^n \to \{0,1\}$, we say that h_n is a distinguisher for W_n if

$$\left| \Pr_{w \sim W_n}[h_n(w) = 1] - \Pr_{x \sim \mathcal{U}_n}[h_n(x) = 1] \right| \geq 1/4.$$

We say that a circuit D_n is a circuit distinguisher for W_n if D_n computes a function h_n that is a distinguisher for W_n. A function $f\colon \{0,1\}^* \to \{0,1\}$ is a distinguisher for a sequence of distributions $\{W_n\}$ if for each large enough n, f_n is a distinguisher for W_n, where f_n is the restriction of f to n-bit inputs.

The following is a slight variant of a definition in [19].

▶ **Definition 15** (Black-Box Generator)**.** Let $\ell\colon \mathbb{N} \to \mathbb{N}$, $\gamma(n) \in [0,1]$, and $\mathfrak{C}$ be a circuit class. A black-box (γ, ℓ)-function generator within $\mathfrak{C}$ is a mapping that associates to any $f\colon \{0,1\}^n \to \{0,1\}$ a family $\mathtt{GEN}(f) = \{g_z\}_{z \in \{0,1\}^m}$ of functions $g_z\colon \{0,1\}^\ell \to \{0,1\}$, for which the following conditions hold:

- **(i)** *Family size.* The parameter $m \leq \mathsf{poly}(n, 1/\gamma)$.
- **(ii)** *Complexity.* For every $z \in \{0,1\}^m$, we have $g_z \in \mathfrak{C}^f[\mathsf{poly}(m)]$.
- **(iii)** *Reconstruction.* Let $L = 2^\ell$ and W_L be the distribution supported over $\{0,1\}^L$ that is generated by $\mathtt{tt}(g_z)$, where $z \sim \mathcal{U}_m$. There is a randomized algorithm A^f, taking as input a circuit D and having oracle access to f, which when D is a distinguishing circuit for W_L, with probability at least $1 - 1/n$ outputs a circuit of size $\mathsf{poly}(n, 1/\gamma, \mathtt{size}(D))$ that is γ-close to f. Furthermore, A^f runs in time at most $\mathsf{poly}(n, 1/\gamma, L(n))$.

This definition is realized by the following result.

▶ **Theorem 16** (Black-Box Generators for Restricted Classes [19]). *Let p be a fixed prime, and $\mathfrak{C}$ be a typical circuit class containing $\mathsf{AC}^0[p]$. For every $\gamma\colon \mathbb{N} \to [0,1]$ and $\ell\colon \mathbb{N} \to \mathbb{N}$ there exists a black-box (γ, ℓ)-function generator within $\mathfrak{C}$.*

▶ **Definition 17** (Complexity Distinguisher). Let $\mathfrak{C}$ be a circuit class and consider functions $s, T\colon \mathbb{N} \to \mathbb{N}$. We say that a probabilistic oracle algorithm A^g is a complexity distinguisher for $\mathfrak{C}[s(n)]$ running in time T if $A^g(1^n)$ always halts in time $T(n)$ with an ouput in $\{0,1\}$, and the following hold:

- For every $g \in \mathfrak{C}[s(n)]$, $\Pr_A[A^g(1^n) = 1] \leq 1/3$.
- $\mathbb{E}_{g \sim \mathcal{F}_n, A}[A^g(1^n)] \geq 2/3$.

▶ **Definition 18** (Zero-Error Complexity Distinguisher). Let $\mathfrak{C}$ be a circuit class and $s, T\colon \mathbb{N} \to \mathbb{N}$ be functions. We say that a probabilistic oracle algorithm A^g is a zero-error complexity distinguisher for $\mathfrak{C}[s(n)]$ running in time T if $A^g(1^n)$ always halts in time $T(n)$ with an output in $\{0,1,?\}$, and the following hold:

- If $g \in \mathfrak{C}[s(n)]$, $A^g(1^n)$ always outputs 0 or ?, and $\Pr_A[A^g(1^n) = ?] \leq 1/3$.
- For every $n \geq 1$ there exists a family of functions $\mathcal{S}_n \subseteq \mathcal{F}_n$ with $|\mathcal{S}_n|/|\mathcal{F}_n| \geq 1 - o(1)$ such that for every $f \in \mathcal{S}_n$, $A^f(1^n)$ always outputs 1 or ?, and $\Pr_A[A^f(1^n) = ?] \leq 1/3$.

We will make use of the following standard concentration of measure result.

▶ **Lemma 19** (Chernoff Bound, cf. [36, Theorem 2.1]). *Let $X \sim \mathsf{Bin}(m,p)$ and $\lambda = mp$. For any $t \geq 0$,*

$$\Pr[|X - \mathbb{E}[X]| \geq t] \leq \exp\left(-\frac{t^2}{2(\lambda + t/3)}\right).$$

3 Learning Speedups and Equivalences

3.1 The Speedup Lemma

We start with the observation that the usual upper bound on the number of small Boolean circuits also holds for unbounded fan-in circuit classes with additional types of gates.

▶ **Lemma 20** (Bound on the number of functions computed by small circuits). *Let $\mathfrak{C}$ be a typical circuit class. For any $s\colon \mathbb{N} \to \mathbb{N}$ satisfying $s(n) \geq n$ there are at most $2^{50 s(n) \log s(n)}$ functions in $\mathcal{F}_n$ computed by $\mathfrak{C}$-circuits of size at most $s(n)$.*

Proof. A circuit over n input variables and of size at most $s(n)$ can be represented by its underlying directed graph together with information about the type of each gate. A node of the graph together with its gate type can be described using $O(\log s(n))$ bits, since for a typical circuit class there are finitely many types of gates. In addition, each input variable can be described as a node of the graph using $O(\log n) = O(\log s(n))$ bits, since by assumption $s(n) \geq n$. Finally, using this indexing scheme, each wire of the circuit corresponding to a directed edge in the circuit graph can be represented with $O(\log s(n))$ bits. Consequently, as we measure circuit size by number of wires, any circuit of size at most $s(n)$ can be represented using at most $O(s(n) \log s(n))$ bits. The lemma follows from the trivial fact that a Boolean circuit computes at most one function in $\mathcal{F}_n$ and via a conservative estimate for the asymptotic notation. ◀

Lemmas 19 and 20 easily imply the following (folklore) result.

▶ **Lemma 21** (Random functions are hard to approximate). *Let $\mathfrak{C}$ be a typical circuit class, $s \geq n$, and $\delta \in [0, 1/2]$. Then,*

$$\Pr_{f \sim \mathcal{F}_n}[\exists\, \mathfrak{C}\text{-circuit of size } \leq s(n) \text{ computing } f \text{ with advantage } \delta(n)] \leq \exp\left(-\delta^2 2^{n-1} + 50 s \log s\right).$$

Proof. Let $g \in \mathcal{F}_n$ be a fixed function. It follows from Lemma 19 with $p = 1/2$, $m = 2^n$, $t = \delta 2^n$, and using $\delta \leq 1/2$ that

$$\Pr_{f \sim \mathcal{F}_n}[g \text{ computes } f \text{ with advantage } \delta(n)] \leq \exp\left(-\frac{\delta^2 2^n}{2}\right).$$

The claim follows immediately from this estimate, Lemma 20, and a union bound. ◀

▶ **Lemma 22** (Non-trivial learners imply distinguishers). *Let $\mathfrak{C}$ be a typical circuit class, $s\colon \mathbb{N} \to \mathbb{N}$ be a size bound, and $T\colon \mathbb{N} \to \mathbb{N}$ be a time bound such that $T(n) = 2^n/n^{\omega(1)}$. If $\mathfrak{C}[s]$ has weak learners running in time T, then $\mathfrak{C}[s(n)]$ has complexity distinguishers running in time $T(n) \cdot \mathsf{poly}(n)$.*

Proof. By the assumption that $\mathfrak{C}$ is weakly learnable, there is a probabilistic oracle algorithm A^f_{learn}, running in time $T(n)$ on input 1^n, which when given oracle access to $f \in \mathfrak{C}[s]$, outputs with probability at least $1 - 1/n$ a Boolean circuit h which agrees with f on at least a $1/2 + 1/n^k$ fraction of inputs of length n, for some universal constant k. We show how to construct from A^f_{learn} an oracle algorithm A^f_{dist} which is a complexity distinguisher for $\mathfrak{C}[s]$.

A^f_{dist} operates as follows on input 1^n. It runs A^f_{learn} on input 1^n. If A^f_{learn} does not output a hypothesis, A^f_{dist} outputs '1'. Otherwise A^f_{dist} estimates the agreement between the hypothesis h output by the learning algorithm and the function f by querying f on n^{5k} inputs of length n chosen uniformly at random, and checking for each such input whether f agrees with h. The estimated agreement is computed to be the fraction of inputs on which f agrees with h. If it is greater than $1/2 + 1/n^{2k}$, A^f_{dist} outputs '0', otherwise it outputs '1'.

By the assumption on efficiency of the learner A^f_{learn}, it follows that A^f_{dist} runs in time $T(n) \cdot \mathsf{poly}(n)$. Thus we just need to argue that A^f_{dist} is indeed a complexity distinguisher.

For a uniformly random f, the probability that A^f_{learn} outputs a hypothesis h that has agreement greater than $1/2 + 1/n^{4k}$ with f is exponentially small. This is because A^f_{learn} runs in time $T(n) = 2^n/n^{\omega(1)}$, and hence if it outputs a hypothesis, it must be of size at most $2^n/n^{\omega(1)}$. By Lemma 21, only an exponentially small fraction of functions can be approximated by circuits of such size. Also, given that a circuit h has agreement at most $1/2 + 1/n^{4k}$ with f, the probability that the estimated agreement according to the procedure above is greater than $1/2 + 1/n^{2k}$ is exponentially small by Lemma 19. Thus, for a uniformly random f, the oracle algorithm A^f_{dist} outputs '0' with exponentially small probability, and hence for large enough n, it outputs '1' with probability at least $2/3$.

For $f \in \mathfrak{C}[s(n)]$, by the correctness and efficiency of the learning algorithm, A^f_{learn} outputs a hypothesis h with agreement at least $1/2 + 1/n^k$ with f, with probability at least $1 - 1/n$. For such a hypothesis h, using Lemma 19 again, the probability that the estimated agreement is smaller than $1/2 + 1/n^{2k}$ is exponentially small. Thus, for n large enough, with probability at least $2/3$, A^f_{dist} outputs '0'. ◀

▶ **Lemma 23** (Faster learners from distinguishers). *Let $\mathfrak{C}$ be a typical circuit class. If $\mathfrak{C}[\mathsf{poly}(n)]$ has complexity distinguishers running in time $2^{O(n)}$, then for every $\varepsilon > 0$, $\mathfrak{C}[\mathsf{poly}(n)]$ has strong learners running in time $O(2^{n^\varepsilon})$. If for some $\varepsilon > 0$, $\mathfrak{C}[2^{n^\varepsilon}]$ has complexity distinguishers running in time $2^{O(n)}$, then $\mathfrak{C}[\mathsf{poly}(n)]$ has strong learners running in time $2^{\log(n)^{O(1)}}$.*

Proof. We prove the first part of the Lemma, and the second part follows analogously using a different parameter setting.

Let $\mathfrak{C}$ be a typical circuit class. If $\mathfrak{C} = \mathsf{AC}^0$, the lemma holds unconditionally since this class can be learned in quasi-polynomial time [46]. Assume otherwise that $\mathfrak{C}$ contains $\mathsf{AC}^0[p]$, for some fixed prime p. By assumption, $\mathfrak{C}[\mathsf{poly}(n)]$ has a complexity distinguisher A_0^g running in time $2^{O(n)}$. We show that for every $\varepsilon > 0$ and every $k > 0$, $\mathfrak{C}[\mathsf{poly}(n)]$ has $(1/n^k, 1/n)$-learners running in time $O(2^{n^\varepsilon})$. Let $\varepsilon' > 0$ be any constant such that $\varepsilon' < \varepsilon$. By Theorem 16 there exists a black-box (γ, ℓ)-function generator $\mathtt{GEN}$ within $\mathfrak{C}$, where $\gamma = 1/n^k$ and $\ell = n^{\varepsilon'}$. For this setting of γ and ℓ we have that the parameter m for $\mathtt{GEN}(f)$ in Definition 15 is $\mathsf{poly}(n)$, and that for each $z \in \{0,1\}^m$, we have $g_z \in \mathfrak{C}^f[\mathsf{poly}(n)]$. Let A_1^f be the randomized reconstruction algorithm for $\mathtt{GEN}(f)$.

We define a $(1/n^k, 0.99)$-learner A^f for $\mathfrak{C}[\mathsf{poly}(n)]$ running in time $O(2^{n^\varepsilon})$; the confidence can then be amplified to satisfy the definition of a strong learner while not increasing the running time of the learner by more than a polynomial factor. The learning algorithm operates as follows. It interprets the oracle algorithm A_0^g on input 1^ℓ as a probabilistic polynomial-time algorithm $D(\cdot, \vec{r})$ which is explicitly given the truth table of g, of size $L = 2^\ell$, as input, with $\vec{r}$ the randomness for this algorithm. It guesses $\vec{r}$ at random and then computes a circuit D_L of size $2^{O(\ell)}$ which is equivalent to $D(\cdot, \vec{r})$ on inputs of size 2^ℓ, using the standard transformation of polynomial-time algorithms into circuits. It then runs A_1^f on input D_L, and halts with the same output as A_1^f. Observe that the queries made by the reconstruction algorithm can be answered by the learner, since it also has query access to f.

Using the bounds on running time of A_0 and A_1, it is easy to see that A^f can be implemented to run in time $2^{O(\ell)}$, which is at most 2^{n^ϵ} for large enough n. We need to argue that A^f is a correct strong learner for $\mathfrak{C}[\mathsf{poly}(n)]$. The critical point is that when $f \in \mathfrak{C}[\mathsf{poly}(n)]$, with noticeable probability, D_L is a distinguishing circuit for W_L (using the terminology of Definition 15), and we can then take advantage of the properties of the reconstruction algorithm. We now spell this out in more detail.

When $f \in \mathfrak{C}[\mathsf{poly}(n)]$, using the fact that $\mathfrak{C}$ is typical and thus closed under composition with itself, and that it contains $\mathsf{AC}^0[p]$, we have that for each $z \in \{0,1\}^m$, $g_z \in \mathfrak{C}[\mathsf{poly}(n)]$. Note that the input size for g_z is $\ell = n^{\varepsilon'}$, and hence also $g_z \in \mathfrak{C}[\mathsf{poly}(\ell)]$. Using now that A_0 is a complexity distinguisher, we have that for any $z \in \{0,1\}^m$, $\Pr_A[A^{g_z}(1^\ell) = 1] \leq 1/3$, while $\mathbb{E}_{g \sim \mathcal{F}_\ell, A}[A^g(1^\ell)] \geq 2/3$. By a standard averaging argument and the fact that probabilities are bounded by 1, this implies that with probability at least 0.05 over the choice of $\vec{r}$, D_L is a distinguishing circuit for W_L. Under the properties of the reconstruction algorithm A_1^f, when given as input such a circuit D_L, with probability at least $1 - 1/n$, the output of A_1^f is $1/n^k$-close to f. Hence with probability at least $0.05 \cdot (1 - 1/n) > 0.01$ over the randomness of A, the output of A^f is $1/n^k$-close to f, as desired. As observed before, the success probability of the learning algorithm can be amplified by standard techniques (cf. [41]).

The second part of the lemma follows by the same argument with a different choice of parameters, using a black-box (γ, ℓ)-function generator with $\gamma = 1/n^k$ and $\ell = (\log n)^c$, where c is chosen large enough as a function of ε. Again, the crucial point is that the relative circuit size of each g_z compared to its number of input bits is within the size bound of the distinguisher. ◀

▶ Remark. While Lemma 23 is sufficient for our purposes, we observe that the same argument shows in fact that the conclusion holds under the weaker assumption that the complexity distinguisher runs in time 2^{n^c}, for a fixed $c \in \mathbb{N}$. In other words, it is possible to obtain faster learners from complexity distinguishers running in time that is quasi-polynomial in the length of the truth-table of its oracle function.

▶ **Lemma 24** (Speedup Lemma). *Let $\mathfrak{C}$ be a typical circuit class. The following hold*:

- (Low-End Speedup) $\mathfrak{C}[\mathsf{poly}(n)]$ *has non-trivial learners if and only if for each $\varepsilon > 0$, $\mathfrak{C}[\mathsf{poly}(n)]$ has strong learners running in time $O(2^{n^\varepsilon})$.*
- (High-End Speedup) *There exists $\varepsilon > 0$ such that $\mathfrak{C}[2^{n^\varepsilon}]$ has non-trivial learners if and only if $\mathfrak{C}[\mathsf{poly}(n)]$ has strong learners running in time $2^{\mathsf{log}(n)^{O(1)}}$.*

Proof. First we show the Low-End Speedup result. The "if" direction is trivial, so we only need to consider the "only if" case. This follows from Lemma 23 and Lemma 22. Indeed, by Lemma 22, if $\mathfrak{C}[\mathsf{poly}(n)]$ has non-trivial learners, it has complexity distinguishers running in time $2^n/n^{\omega(1)}$. By Lemma 23, the existence of such complexity distinguishers implies that for each $\varepsilon > 0$, $\mathfrak{C}[\mathsf{poly}(n)]$ has strong learners running in time $O(2^{n^\varepsilon})$, and we are done.

Next we show the High-End Speedup result. The proof for the "only if" direction is completely analogous to the corresponding proof for the Low-End Speedup result. The "if" direction, however, is not entirely trivial. We employ a standard padding argument to establish this case, thus completing the proof of Lemma 24.

Suppose that $\mathfrak{C}[\mathsf{poly}(n)]$ has a strong learner running in time $2^{\mathsf{log}(n)^c}$, for some constant $c > 0$. Let A_{low} be a learning algorithm witnessing this fact. We show how to use A_{low} to construct a learning algorithm A_{high} which is a $(1/n, 1/\mathsf{poly}(n))$-learner for $\mathfrak{C}[2^{n^{1/3c}}]$, and runs in time $\leq 2^{\sqrt{n}}$. As usual, confidence can be boosted without a significant increase of running time, and it follows that $\mathfrak{C}[2^{n^{1/3c}}]$ has non-trivial learners according to our definition.

On input 1^n and with oracle access to some function $f\colon \{0,1\}^n \to \{0,1\}$, $A^f_{\mathsf{high}}(1^n)$ simulates $A^{f'}_{\mathsf{low}}(1^{n'})$, where $n' \stackrel{\text{def}}{=} n + 2^{\lceil n^{1/3c} \rceil}$, and $f'\colon \{0,1\}^{n'} \to \{0,1\}$ is the (unique) Boolean function satisfying the following properties. For any input $x' = xy \in \{0,1\}^{n'}$, where $|y| = 2^{\lceil n^{1/3c} \rceil}$ and $|x| = n$, $f'(x')$ is defined to be $f(x)$. Note that if $f \in \mathfrak{C}[2^{n^{1/3c}}]$ then $f' \in \mathfrak{C}[O(n')]$: the linear-size $\mathfrak{C}$-circuit for f' on an input x' of length n' just simulates the smallest $\mathfrak{C}$-circuit for f on its n-bit prefix. During the simulation, whenever A_{low} makes an oracle call x' to f', A_{high} answers it using an oracle call x to f, where x is the prefix of x' of length n. By definition of f', this simulation step is always correct. When $A^{f'}_{\mathsf{low}}$ completes its computation and outputs a hypothesis h' on n' input bits, A_{high} outputs a modified hypothesis h as follows: it chooses a random string r of length $2^{\lceil n^{1/3c} \rceil}$, and outputs the circuit h_r defined by $h_r(x) \stackrel{\text{def}}{=} h'(xr)$. Note that by the assumed efficiency of A_{low}, A_{high} halts in time $\leq 2^{\sqrt{n}}$ on large enough n.

By the discussion above, it is enough to argue that the hypothesis h output by A_{high} is a good hypothesis with probability at least $1/\mathsf{poly}(n)$. Since A_{low} is a strong learner and since the f' used as oracle to A_{low} in the simulation has linear size, for all large enough n, with probability at least $1 - 1/n'$, h' disagrees with f' on at most a $1/(n')^k$ fraction of inputs of length n', where k is a large enough constant fixed in the construction above. Consider a randomly chosen r of length $n' - n$. By a standard Markov-type argument, when h' is good, for at least a $1/\mathsf{poly}(n)$ fraction of the strings r, $h_r(x)$ disagrees with $f(x)$ on at most a $1/n$ fraction of inputs. This completes the argument. ◀

3.2 Equivalences for Learning, Compression, and Distinguishers

▶ **Theorem 25** (Algorithmic Equivalences). *Let $\mathfrak{C}$ be a typical circuit class. The following statements are equivalent*:

1. $\mathfrak{C}[\mathsf{poly}(n)]$ *has non-trivial learners.*
2. *For each $\varepsilon > 0$ and $k \in \mathbb{N}$, $\mathfrak{C}[\mathsf{poly}(n)]$ can be learned to error $\leq n^{-k}$ in time $O(2^{n^\epsilon})$.*
3. $\mathfrak{C}[\mathsf{poly}(n)]$ *has probabilistic (exact) compression.*
4. $\mathfrak{C}[\mathsf{poly}(n)]$ *has probabilistic average-case compression with error $o(1)$.*

5. $\mathfrak{C}[\mathsf{poly}(n)]$ *has complexity distinguishers running in time* $2^{O(n)}$.
6. *For each* $\varepsilon > 0$, $\mathfrak{C}[\mathsf{poly}(n)]$ *has complexity distinguishers running in time* $O(2^{n^\epsilon})$.
7. $\mathfrak{C}[\mathsf{poly}(n)]$ *can be learned using membership and equivalence queries to sub-constant error in non-trivial time.*

Proof. We establish these equivalences via the following complete set of implications:

(5) ⇒ (2): This follows from Lemma 23.

(2) ⇒ (1) and (6) ⇒ (5): These are trivial implications.

(2) ⇒ (4): Probabilistic compression for $\mathfrak{C}[\mathsf{poly}(n)]$ follows from simulating a $(1/n^3, 1/n)$-learner for the class running in time $O(2^{\sqrt{n}})$, and answering any oracle queries by looking up the corresponding bit in the truth table of the function, which is given as input to the compression algorithm. The compression algorithm returns as output the hypothesis of the strong learner, and by assumption this agrees on a $(1 - 1/n^3)$ fraction of inputs of length n with the input function, with probability at least $1 - 1/n$. Moreover, since the simulated learner runs in time $O(2^{\sqrt{n}})$, the circuit that is output has size at most $O(2^{\sqrt{n}})$. It is clear that the simulation of the learner can be done in time $2^{O(n)}$, as required for a compression algorithm.

(2) ⇒ (3): This follows exactly as above, except that there is an additional step after the simulation of the learner. Once the learner has output a hypothesis h, the compression algorithm compares this hypothesis with its input truth table entry by entry, simulating h whenever needed. If h differs from the input truth table on more than a $1/n^3$ fraction of inputs, the compression algorithm rejects – this happens with probability at most $1/n$ by assumption on the learner. If h and the input truth table differ on at most $1/n^3$ fraction of inputs of length n, the compression algorithm computes by brute force a circuit of size at most $2^n/n^2$ which computes the function h' that is the XOR of h and the input truth table. The upper bound on size follows from the fact that h' has at most $2^n/n^3$ 1's. Finally, the compression algorithm outputs $h \oplus h'$. For any typical circuit class, the size of the corresponding circuit is $O(2^n/n^2)$. Note that $h \oplus h'$ computes the input truth table exactly.

(2) ⇒ (6): This follows from Lemma 22.

(1) ⇒ (5), (3) ⇒ (5), and (4) ⇒ (5): The distinguisher runs the circuit output by the learner or compression algorithm on every input of length n, and computes the exact agreement with its input f on length n by making 2^n oracle queries to f. If the circuit agrees with f on at least a 2/3 fraction of inputs, the distinguisher outputs 0, otherwise it outputs 1. By the assumption on the learner/compression algorithm, for $f \in \mathfrak{C}[\mathsf{poly}(n)]$, the distinguisher outputs 0 with probability at least 2/3. Using Lemma 21, for a random function, the probability that the distinguisher outputs 1 is at least 2/3.

(7) ⇒ (5): The complexity distingisher has access to the entire truth-table, and can answer the membership and equivalence queries of the learner in randomized time $2^{O(n)}$. Randomness is needed only to simulate the random choices of the learning algorithm, while the answer to each query can be computed in deterministic time. Since the learner runs in time $2^n/n^{\omega(1)}$, whenever it succeeds it outputs a hypothesis circuit of at most this size. The complexity distinguisher can compare this hypothesis to its input truth-table, and similarly to the arguments employed before, is able to distinguish random functions from functions in $\mathfrak{C}[\mathsf{poly}(n)]$.

(2) ⇒ (7): This is immediate since the algorithm from (2) is faster, has better accuracy, and makes no equivalence queries. ◀

▶ **Theorem 26** (Equivalences for zero-error algorithms). *Let* $\mathfrak{C}[\mathsf{poly}(n)]$ *be a typical circuit class. The following statements are equivalent*:

1. *There are* P*-natural proofs useful against* $\mathfrak{C}[\mathsf{poly}(n)]$.
2. *There are* ZPP*-natural proofs useful against* $\mathfrak{C}[\mathsf{poly}(n)]$.
3. *For each* $\varepsilon > 0$*, there are* $\mathsf{DTIME}(O(2^{(\log N)^{\varepsilon}}))$*-natural proofs useful against* $\mathfrak{C}[\mathsf{poly}(n)]$*, where* $N = 2^n$ *is the truth-table size.*
4. *For each* $\varepsilon > 0$*, there are zero-error complexity distinguishers for* $\mathfrak{C}[\mathsf{poly}(n)]$ *running in time* $O(2^{n^{\varepsilon}})$.

Proof. We establish these equivalences via the following complete set of implications:

(1) $\Rightarrow$ (2): This is a trivial implication.

(3) $\Rightarrow$ (4): This is almost a direct consequence of the definitions, except that the density of the natural property has to be amplified to $1 - o(1)$ before converting the algorithm into a zero-error complexity distinguisher. This is a standard argument, and can be achieved by defining a new property from the initial one. More details can be found, for instance, in the proof of [19, Lemma 2.7].

(2) $\Rightarrow$ (1):[10] Let A be an algorithm running in zero-error probabilistic time m^k on inputs of length m and with error probability $\leq 1/4$, for m large enough and k an integer, and deciding a combinatorial property $\mathfrak{R}$ useful against $\mathfrak{C}[\mathsf{poly}(n)]$. We show how to define a combinatorial property $\mathfrak{R}'$ useful against $\mathfrak{C}[\mathsf{poly}(n)]$ such that $\mathfrak{R}' \in \mathsf{P}$, and such that at least a $1/8$ fraction of the truth tables of any large enough input length belong to $L_{\mathfrak{R}'}$. This fraction can be amplified by defining a new natural property $\mathfrak{R}''$ such that any string yz with $|y| = |z|$ belongs to $L_{\mathfrak{R}''}$ if and only if either $y \in L_{\mathfrak{R}'}$ or $z \in L_{\mathfrak{R}'}$ (see e.g. [19]).

We define $\mathfrak{R}'$ via a deterministic polynomial-time algorithm A' deciding $L_{\mathfrak{R}'}$. Given an input truth table y of size $2^{n'}$, A' acts as follows: it determines the largest integer n such that $n(k+1) < n'$. It decomposes the input truth table as $y = xzw$, where $|x| = 2^n$, $|z| = 2^{kn}$, and the remaining part w is irrelevant. It runs A on x, using z as the randomness for the simulation of A. If A accepts, it accepts; if A rejects or outputs '?', it rejects.

It should be clear that A' runs in polynomial time. The fact that A' accepts at least a $1/8$ fraction of truth tables of any large enough input length follows since for any $x \in L_{\mathfrak{R}}$, A outputs '?' with probability at most $1/3$, and at least a $1/2$ fraction of strings of length 2^n are in $L_{\mathfrak{R}}$. It only remains to argue that the property $\mathfrak{R}'$ is useful against $\mathfrak{C}[\mathsf{poly}(n)]$. But any string y of length $2^{n'}$ accepted by A' has as a substring the truth table of a function on $n = \Omega(n')$ bits which is accepted by A and hence is in $L_{\mathfrak{R}}$. Since $\mathfrak{R}$ is useful against $\mathfrak{C}[\mathsf{poly}(n)]$, this implies that $\mathfrak{R}'$ is useful against $\mathfrak{C}[\mathsf{poly}(n)]$.

(4) $\Rightarrow$ (3): The proof is analogous to (2) $\Rightarrow$ (1).

(3) $\Rightarrow$ (1): This is a trivial direction since $N = 2^n$.

(1) $\Rightarrow$ (3): This implication uses an idea of Razborov and Rudich [59]. Suppose there are P-natural proofs useful against $\mathfrak{C}[\mathsf{poly}(n)]$. This means in particular that for every $c \geq 1$, there is a polynomial-time algorithm A_c, which on inputs of length 2^n, where $n \in \mathbb{N}$, accepts at least a $1/2$ fraction of inputs, and rejects all inputs y such that $\mathtt{fn}(y) \in \mathfrak{C}[n^c]$. Consider an input y to A_c of length 2^n, and let $\varepsilon > 0$ be fixed. Let y' be the substring of y such that $\mathtt{fn}(y')$ is the subfunction of $\mathtt{fn}(y)$ obtained by fixing the first $n - n^{\varepsilon}$ bits of the input to $\mathtt{fn}(y)$ to 0. It is easy to see that if $\mathtt{fn}(y) \in \mathfrak{C}[n^c]$, then $\mathtt{fn}(y') \in \mathfrak{C}[(n')^{c/\varepsilon}]$, where n' denotes the number of input bits of $\mathsf{fn}(y')$.

Let $d \geq 1$ be any constant, and $\varepsilon > 0$ be fixed. We show how to define an algorithm B_d which runs in time $O(2^{\mathsf{log}(N)^{\varepsilon}})$ on an input of length $N = 2^n$, deciding a combinatorial property which is useful against $\mathfrak{C}$-circuits of size n^d. (Using the same approach, it is possible

[10] This argument is folklore. It has also appeared in more recent works, such as [74].

to design a single algorithm that works for any fixed d whenever n is large enough, provided that we start with a natural property that is useful in this stronger sense.) On input y of length N, B_d computes y' of length $2^{\log(N)^\varepsilon}$, as defined in the previous paragraph. For the standard encoding of truth tables, y' is a prefix of y, and can be computed in time $O(|y'|)$. B_d then simulates $A_{\lceil d/\varepsilon \rceil}$ on y', accepting if and only if the simulated algorithm accepts. The simulation halts in time $\mathsf{poly}(|y'|)$, as $A_{\lceil d/\varepsilon \rceil}$ is a poly-time algorithm. For a random input y, B_d accepts with probability at least $1/2$, using that y' is uniformly distributed, and the assumption that $A_{\lceil d/\varepsilon \rceil}$ witnesses natural proofs against a circuit class. For an input y such that $\mathtt{fn}(y) \in \mathfrak{C}[n^d]$, B_d always rejects, as in this case, $\mathtt{fn}(y') \in \mathfrak{C}[(n')^{d/\varepsilon}]$, and so $A_{\lceil d/\varepsilon \rceil}$ rejects y', using the assumption that $A_{\lceil d/\varepsilon \rceil}$ decides a combinatorial property useful against n-bit Boolean functions in $\mathfrak{C}[n^{\lceil d/\varepsilon \rceil}]$. ◀

4 Learning versus Pseudorandom Functions

4.1 The PRF-Distinguisher Game

In this section we consider (non-uniform) randomized oracle circuits $B^{\mathcal{O}}$ from $\mathsf{Circuit}^{\mathcal{O}}[t]$, where t is an upper bound on the number of wires in the circuit. Recall that a circuit from this class has a special gate type that computes according to the oracle $\mathcal{O}$, which will be set to some fixed Boolean function $f\colon \{0,1\}^m \to \{0,1\}$ whenever we discuss the computation of the circuit.

We will view such circuits either as distinguishers or learning algorithms, where the oracle is the primary input to the circuit. For this reason and because our results are stated in the non-uniform setting, we assume from now on that such circuits have no additional input except for variables $y_1, \ldots, y_\ell$ representing the random bits, where $\ell \leq t$. If $w \in \{0,1\}^\ell$ is a fixed sequence of bits, we use $B^{\mathcal{O}}_w$ to denote the deterministic oracle circuit obtaining from the circuit $B^{\mathcal{O}}$ by setting its randomness to w. Observe that (non-uniform) learning algorithms can be naturally described by randomized oracle circuits from $\mathsf{Circuit}^{\mathcal{O}}$ with multiple output bits. The output bits describe the output hypothesis, under some reasonable encoding for Boolean circuits.[11]

We will consider pairs $(G_n, \mathcal{D}_n)$ where $G_n \subseteq \mathcal{F}_n$ and $\mathcal{D}_n$ is a distribution with $\mathsf{Support}(\mathcal{D}_n) \subseteq G_n$. This notation is convenient when defining samplable function families and pseudorandom function families.

▶ **Definition 27** (Pseudorandom Function Families)**.** We say that a pair $(G_n, \mathcal{D}_n)$ is a $(t(n), \varepsilon(n))$-*pseudorandom function family (PRF)* in $\mathcal{C}[s(n)]$ if $G_n \subseteq \mathcal{C}[s(n)]$ and for every randomized oracle circuit $B^{\mathcal{O}} \in \mathsf{Circuit}^{\mathcal{O}}[t(n)]$,

$$\left| \Pr_{g \sim \mathcal{D}_n,\, w}[B^g(w) = 1] - \Pr_{f \sim \mathcal{F}_n,\, w}[B^f(w) = 1] \right| \leq \varepsilon.$$

This definition places no constraint on the complexity of generating the pair $(G_n, \mathcal{D}_n)$. In order to capture this, we restrict attention to $G_n \subseteq \mathcal{C}_n$ for some typical circuit class $\mathfrak{C} = \{\mathcal{C}_n\}$, and assume a fixed encoding of circuits from $\mathfrak{C}$ by strings of length polynomial in the size of the circuit. We say that a circuit $A \in \mathsf{Circuit}[S]$ is a $\mathcal{C}_n$-*sampler* if A outputs valid descriptions of circuits from $\mathcal{C}_n$.

[11] In this non-uniform framework it is possible to derandomize a learning circuit with some blow-up in circuit size, but we will not be concerned with this matter here.

▶ **Definition 28** (Samplable Function Families)**.** We say that a pair $(G_n, \mathcal{D}_n)$ with $G_n \subseteq \mathcal{C}_n$ is *S-samplable* if there exists a $\mathcal{C}_n$-sampler $A \in \mathsf{Circuit}[S]$ on $\ell \leq S$ input variables such that $A(\mathcal{U}_\ell) \equiv \mathcal{D}_n$, where we associate each output string of A to its corresponding Boolean function.

It is well-known that the existence of learning algorithms for a circuit class $\mathcal{C}_n[s(n)]$ implies that there are no secure pseudorandom function families in $\mathcal{C}_n[s(n)]$. Moreover, this remains true even for function families that are not efficiently samplable. Following the notation from Definition 7, we can state a particular form of this observation as follows.

▶ **Proposition 29** (Learning $\mathfrak{C}$ implies no PRFs in $\mathfrak{C}$)**.** *Assume there is a randomized oracle circuit in $\mathsf{Circuit}^{\mathcal{O}}[t(n)]$ that $(1/3, 1/n)$-learns every function in $\mathcal{C}_n[s(n)]$, where $n \leq t(n) \leq 2^n/n^2$. Then for large enough n there are no $(\mathtt{poly}(t(n)), 1/10)$-pseudorandom function families in $\mathcal{C}_n[s(n)]$.*

Our goal for the rest of this section is to establish a certain converse of Proposition 29 (Theorem 34 and Corollary 35). An important technical tool will be a "small-support" version of the min-max theorem, described next.

Small-Support Approximate Min-Max Theorem for Bounded Games [9, 47]. We follow the notation from [47]. Let M be an $r \times c$ real-valued matrix, p be a probability distribution over its rows, and q be a probability distribution over its columns. The classic min-max theorem [66] states that

$$\min_{p} \max_{j \in [c]} \mathbb{E}_{i \sim p}[M(i,j)] \;=\; \max_{q} \min_{i \in [r]} \mathbb{E}_{j \sim q}[M(i,j)]. \tag{1}$$

The distributions p and q are called *mixed strategies*, while individual indexes i and j are called *pure strategies*. We use $v(M)$ to denote the value in Equation 1. (Recall that this can be interpreted as a game between a row player, or *Minimizer*, and a column player, or *Maximizer*. The min-max theorem states that the order in which the players reveal their strategies does not change the value of the game. It is easy to see that the second player can be restricted to pure strategies.)

We will consider a game played on a matrix of exponential size, and will be interested in near-optimal mixed strategies with succinct descriptions. This motivates the following definitions. A mixed strategy is *k-uniform* if it is selected uniformly from a multiset of at most k pure strategies. We use P_k and Q_k to denote the set of k-uniform strategies for the row player and the column player, respectively. For convenience, given a mixed row strategy p, we let $v(p) = v_M(p) = \max_{j \in [c]} \mathbb{E}_{i \sim p}[M(i,j)]$. Similarly, we use $v(q) = v_M(q) = \min_{i \in [r]} \mathbb{E}_{j \sim q}[M(i,j)]$ for a column mixed strategy q. We say that a mixed strategy u is *δ-optimal* if $|v(u) - v(M)| \leq \delta$.

We will need the following "efficient" version of the min-max theorem.

▶ **Theorem 30** (Small-Support Min-Max Theorem [9, 47])**.** *Let M be a $r \times c$ real-valued matrix with entries in the interval $[-1, 1]$. For every $\delta > 0$, if $k_r \geq 10\ln(c)/\delta^2$ and $k_c \geq 10\ln(r)/\delta^2$ then*

$$\min_{p \in P_{k_r}} v(p) \;\leq\; v(M) + \delta, \quad \textit{and} \quad \max_{q \in Q_{k_c}} v(q) \;\geq\; v(M) - \delta.$$

In other words, there are δ-optimal strategies for the row and column players with relatively small support size.

The PRF-Distinguisher Game. Let $\mathcal{C}_n[s]$ be a circuit class and $\mathsf{Circuit}^{\mathcal{O}}[t]$ be an oracle circuit class, with size parameters $s(n)$ and $t(n)$, respectively. We consider a $[-1,1]$-valued matrix $M = M^{\mathcal{C}_n[s],\mathsf{Circuit}^{\mathcal{O}}[t]}$, defined as follows. The rows of M are indexed by Boolean functions in $\mathcal{C}_n[s]$, and the columns of M are indexed by (single-output) *deterministic* oracle circuits from $\mathsf{Circuit}^{\mathcal{O}}[t]$. In other words, such circuit have access to constants 0 and 1, compute according to the values of the oracle gates, and produce an output value in $\{0,1\}$. In order not to introduce further notation, we make the simplifying assumption that the negation of every circuit from $\mathsf{Circuit}^{\mathcal{O}}[t]$ is also in $\mathsf{Circuit}^{\mathcal{O}}[t]$. For $h \in \mathcal{C}_n[s]$ and $C^{\mathcal{O}} \in \mathsf{Circuit}^{\mathcal{O}}[t]$, we let

$$M(h, C^{\mathcal{O}}) \stackrel{\text{def}}{=} C^h - \Pr_{f \sim \mathcal{F}_n}[C^f = 1],$$

where $C^g \in \{0,1\}$ denotes the output of $C^{\mathcal{O}}$ when computing with oracle $\mathcal{O} = g$, for a fixed $g\colon \{0,1\}^n \to \{0,1\}$. We say that the matrix M is the *PRF-Distinguisher game* for $\mathcal{C}_n[s]$ and $\mathsf{Circuit}^{\mathcal{O}}[t]$. Observe that this is a finite matrix, for every choice of n.

Following our notation, we use $v(M)$ to denote the value of the game corresponding to M, which can be interpreted as follows. Let p be a mixed strategy for the row player. In other words, p is simply a distribution over functions from $\mathcal{C}_n[s]$. Consequently, to each row strategy p we can associate a pair $(G_p, \mathcal{D}_p)$, where $p = \mathcal{D}_p$ and $G_p = \mathsf{Support}(\mathcal{D}_p)$, as in Definition 27. On the other hand, a mixed strategy q over the columns is simply a distribution over deterministic oracle circuits from $\mathsf{Circuits}^{\mathcal{O}}[t]$, which can be interpreted as a (non-constructive) randomized circuit $B^{\mathcal{O}}$. Under this interpretation, the value of the game when played with strategies p and q is given by

$$\begin{aligned}
\mathbb{E}_{h\sim p,\, C^{\mathcal{O}}\sim q}[M(h, C^{\mathcal{O}})] &= \mathbb{E}_{h, C^{\mathcal{O}}}[C^h - \Pr_{f\sim\mathcal{F}_n}[C^f = 1]] \\
&= \mathbb{E}_{h, C^{\mathcal{O}}}[C^h] - \Pr_{f, C^{\mathcal{O}}\sim q}[C^f = 1] \\
&= \Pr_{g\sim\mathcal{D}_p,\, B^{\mathcal{O}}}[B^g = 1] - \Pr_{f\sim\mathcal{F}_n,\, B^{\mathcal{O}}}[B^f = 1],
\end{aligned}$$

which corresponds to the distinguishing probability in Definition 27 without taking absolute values. But since we assumed that the circuits indexing the columns of M are closed under complementation, it follows that the (global) value $v(M)$ of this game captures the security of PRFs from $\mathcal{C}_n[s]$ against $\mathsf{Circuit}^{\mathcal{O}}[t]$-distinguishers. (Notice though that this value does not take into account the samplability of the function families involved, nor the constructivity of the ensemble of distinguishers corresponding to a "randomized" oracle distinguisher in the argument above.)

4.2 A (Non-Uniform) Converse to "Learning Implies no PRFs"

We proceed with our original goal of establishing a converse of Proposition 29. Roughly speaking, we want to show that if every samplable function family from $\mathcal{C}_n$ can be distinguished from a random function (possibly by different distinguishers), then there is a single algorithm that learns every function in $\mathcal{C}_n$. Formally, what we get is a sequence of subexponential size (non-uniform) circuits learning $\mathfrak{C}$.

The proofs of Lemmas 31 and 32 below rely on Theorem 30.

▶ **Lemma 31** ($\nexists$ samplable PRF $\to \nexists$ PRF against ensembles of circuits)**.** *There exists a universal constant $c \in \mathbb{N}$ for which the following holds. Let $t(n) \geq n$, $s(n) \geq n$, $\delta(n) > 0$, and $\varepsilon(n) > 0$ be arbitrary functions. If there is no $O(t \cdot s \cdot 1/\delta)^c$-samplable pair $(\widetilde{G_n}, \widetilde{\mathcal{D}_n})$ that is a $(t(n), \varepsilon(n) + \delta(n))$-PRF in $\mathcal{C}_n[s(n)]$, then there is no pair $(G_n, \mathcal{D}_n)$ with $G_n \subseteq \mathcal{C}_n[s(n)]$ that $\varepsilon(n)$-fools every ensemble of deterministic $\mathsf{Circuit}^{\mathcal{O}}[t(n)]$-circuits.*

Proof. We use Theorem 30 to establish the contrapositive. Assume there exists a pair $(G_n, \mathcal{D}_n)$ where $\mathcal{D}_n$ is distributed over $G_n \subseteq \mathcal{C}_n[s(n)]$ such that for every distribution q over $\mathsf{Circuit}^{\mathcal{O}}[t(n)]$ we have

$$\left| \Pr_{g \sim \mathcal{D}_n, C^{\mathcal{O}} \sim q}[C^g = 1] - \Pr_{f \sim \mathcal{F}_n, C^{\mathcal{O}} \sim q}[C^f = 1] \right| \leq \varepsilon(n).$$

Let $p = \mathcal{D}_n$, and observe that in the corresponding PRF-Distinguisher game we get $v_M(p) \leq \varepsilon(n)$. Consequently, $v(M) \leq \min_p v_M(p) \leq \varepsilon(n)$. It follows from Theorem 30 and a bound on the number of columns of M (similar to Lemma 20) that there exists a k-uniform distribution $\tilde{p}$ over functions in $\mathcal{C}_n[s(n)]$ with $k \leq O(\ln 2^{O(t \log t)}/\delta(n)^2) = O((t \log t)/\delta(n)^2)$ such that $v_M(\tilde{p}) \leq \varepsilon(n) + \delta(n)$.

In other words, each $f \in \mathsf{Support}(\tilde{p})$ is in $\mathcal{C}_n[s(n)]$, the support of this distribution contains at most $O((t \log t)/\delta(n)^2)$ different functions, and each such function can be encoded by a string of length $\mathtt{poly}(s(n))$ that describes the corresponding circuit. Using that $\tilde{p}$ is a k-uniform distribution, it is not hard to see that there exists a circuit $A \in \mathsf{Circuit}[S]$ with $A(\mathcal{U}_\ell) \equiv \tilde{p}$ for some $\ell \leq S$, where $S \leq \mathtt{poly}(t, s, 1/\delta)$. Since every randomized circuit $B^{\mathcal{O}}$ can be seen as a distribution over deterministic oracle circuits, it follows that there is an S-samplable pair $(\widetilde{G_n}, \widetilde{\mathcal{D}_n})$ that is a $(t(n), \varepsilon(n) + \delta(n))$-PRF in $\mathcal{C}_n[s(n)]$. This completes the proof. ◀

▶ **Lemma 32** (∄ PRF against ensembles of circuits → ∃ universal distinguisher)**.** *There exists a universal constant $c \in \mathbb{N}$ for which the following holds. Let $s(n) \geq n$, $t(n) \geq n$, $\varepsilon(n) > 0$, and $\gamma(n) > 0$ be arbitrary functions. If there is no pair $(G_n, \mathcal{D}_n)$ with $G_n \subseteq \mathcal{C}_n[s(n)]$ that $\varepsilon(n)$-fools every ensemble of deterministic $\mathsf{Circuit}^{\mathcal{O}}[t(n)]$-circuits, then there is a randomized oracle circuit $B^{\mathcal{O}} \in \mathsf{Circuit}^{\mathcal{O}}[O(t \cdot s \cdot 1/\gamma)^c]$ that distinguishes every such pair from a random function with advantage at least $\varepsilon(n) - \gamma(n)$.*

Proof. We rely on the classical min-max theorem and on Theorem 30. It follows from the assumption of the lemma that the corresponding PRF-Distinguisher game has value $v(M) \geq \varepsilon(n)$. By the min-max theorem, there is an ensemble of $\mathsf{Circuit}^{\mathcal{O}}[t(n)]$-circuits that distinguishes every pair $(G_n, \mathcal{D}_n)$ satisfying $G_n \subseteq \mathcal{C}_n[s(n)]$ with advantage at least $\varepsilon(n)$. Applying Theorem 30, we obtain a k-uniform distribution q over deterministic $\mathsf{Circuit}^{\mathcal{O}}[t(n)]$-circuits with distinguishing probability at least $\varepsilon(n) - \gamma(n)$ and support size at most $k = O(\ln 2^{O(s \log s)}/\gamma(n)^2) = O((s \log s)/\gamma(n)^2)$. Similarly to the proof of Lemma 31, this ensemble of circuits implies the existence of a single randomized oracle circuit $B^{\mathcal{O}} \in \mathsf{Circuit}^{\mathcal{O}}[O(s \cdot t \cdot 1/\gamma)^c]$ that distinguishes every pair $(G_n, \mathcal{D}_n)$ with $G_n \subseteq \mathcal{C}_n[s(n)]$ from a random function with advantage at least $\varepsilon(n) - \gamma(n)$. This completes the proof. ◀

Lemmas 31 and 32 hold for each value of n. The next lemma is a reduction involving different values of this parameter.

▶ **Lemma 33** (∃ universal distinguishers → ∃ learning circuits)**.** *Assume that for every $k \geq 1$ and large enough n there exists a randomized oracle circuit $B_n^{\mathcal{O}}$ in $\mathsf{Circuit}^{\mathcal{O}}[2^{O(n)}]$ that distinguishes every pair $(G_n, \mathcal{D}_n)$ with $G_n \subseteq \mathcal{C}_n[n^k]$ from a random function with advantage $\geq 1/40$. Then for every $\ell \geq 1$ and $\varepsilon > 0$ there is a non-uniform sequence of randomized oracle circuits in $\mathsf{Circuit}^{\mathcal{O}}[2^{n^\varepsilon}]$ that learn every function $f \in \mathcal{C}_n[n^\ell]$ to error at most $n^{-\ell}$.*

Proof. This lemma is simply a (weaker) non-uniform version of the proof of Lemma 23 from Section 3. It is enough to use the sequence of randomized oracle circuits $B_n^{\mathcal{O}}$ as distinguishing circuits, and to observe that the statement of Theorem 16 holds with an arbitrarily small constant in the distinguishing probability. ◀

Recall that $\mathfrak{C} = \{\mathcal{C}_n\}$ is an arbitrary typical circuit class. The main technical result of this section follows from Lemmas 31, 32, and 33 together with an appropriate choice of parameters.

▶ **Theorem 34** (No samplable PRFs in $\mathfrak{C}$ implies Learning $\mathfrak{C}$). *If $t(n) \leq 2^{O(n)}$ and $c' \geq 1$ is a large enough constant, the following holds. Suppose that for every $k \geq 1$ each $O((t(n) \cdot n^k)^{c'})$-samplable pair $(G_n, \mathcal{D}_n)$ with $G_n \subseteq \mathcal{C}_n[n^k]$ can be distinguished from a random function by some randomized oracle circuit from $\mathsf{Circuit}^{\mathcal{O}}[t(n)]$ with advantage at least $1/10$. Then, for every $k \geq 1$, $\varepsilon > 0$, and large enough n, there is a randomized oracle circuit from $\mathsf{Circuit}^{\mathcal{O}}[2^{n^\varepsilon}]$ that learns every function in $\mathcal{C}_n[n^k]$ to error at most n^{-k}.*

Proof. The existence of the learning circuit will follow if we can prove that the hypothesis of Lemma 33 is satisfied. Thus it is enough to argue that, for every $k \geq 1$ and large enough n, there is a (single) randomized oracle circuit $B^{\mathcal{O}}$ from $\mathsf{Circuit}^{\mathcal{O}}[2^{O(n)}]$ that distinguishes with advantage $\geq 1/40$ every pair $(G_n, \mathcal{D}_n)$ with $G_n \subseteq \mathcal{C}_n[n^k]$. In turn, this follows from Lemma 32 for $s(n) = n^k$, $\varepsilon(n) = 1/20$, and $\gamma(n) = 1/40$ if there is no pair $(G_n, \mathcal{D}_n)$ with $G_n \subseteq \mathcal{C}_n[n^k]$ that $1/20$-fools every ensemble of deterministic oracle circuits from $\mathsf{Circuit}^{\mathcal{O}}[2^{O(n)}]$, for a slightly smaller constant in the latter exponent. But this is implied by the hypothesis of Theorem 34 together with Lemma 31, instantiated with our value $t(n) \leq 2^{O(n)}$, $s(n) = n^k$, $\varepsilon(n) = 1/20$, and $\delta(n) = 1/20$, provided that we take c' sufficiently large. This completes the proof. ◀

Dropping the samplability condition, we get the following weaker statement, which provides a converse of Proposition 29 in the regime where $t(n)$ is exponential and $s(n)$ is polynomial.

▶ **Corollary 35** (No PRFs in $\mathfrak{C}$ implies Learning $\mathfrak{C}$). *Let $t(n) \leq 2^{O(n)}$. If for every $k \geq 1$ and large enough n there are no $(\mathsf{poly}(t(n)), 1/10)$-pseudorandom function families in $\mathcal{C}_n[n^k]$, then for every $\varepsilon > 0$, $k \geq 1$, and large enough n, there is a randomized oracle circuit in $\mathsf{Circuit}^{\mathcal{O}}[2^{n^\varepsilon}]$ that $(n^{-k}, 1/n)$-learns every function in $\mathcal{C}_n[n^k]$.*

We observe that smaller time bounds $t(n)$ do not necessarily lead to smaller learning circuits, due to the running time of the black-box generator in Definition 15 and Theorem 16. However, a smaller $t(n)$ implies a weaker samplability condition in the statement of Theorem 34, which makes it stronger. A natural question is whether a more efficient distinguisher implies that larger circuits can be distinguished by subexponential size oracle circuits, in analogy to Lemma 24. We mention that no simple reduction via padding seems to work, since a random function on n bits mapped into a larger domain via projections is no longer a uniformly random function. Finally, the distinguishing advantage $1/10$ is arbitrary. Indeed, it can be assumed to be much lower, by following the estimates in the proof of Theorem 16.

▶ Remark. In order to prove Theorem 34, we have made essential use of the "efficient" min-max theorem from [9, 47], which guarantees the existence of near-optimal mixed strategies with simple descriptions. Unfortunately, this result does not provide an efficient algorithm to produce such strategies, which would lead to an equivalence between learning algorithms and the nonexistence of pseudorandom functions with respect to uniform computations. While there are more recent works that explore uniform versions of the min-max theorem (cf. [67]), they assume the existence of certain auxiliary algorithms in order to construct the near-optimal strategies, and it is unclear to us if they can be applied in the context of Theorem 34.

5 Lower Bounds from Nontrivial Algorithms

▶ **Theorem 36** (Circuit lower bounds from nontrivial learning algorithms). *Let $\mathfrak{C}$ be any typical circuit class. If for each k, $\mathfrak{C}[n^k]$ has non-trivial learning algorithms, then for each k,* $\mathsf{BPTIME}(2^{O(n)}) \not\subseteq \mathfrak{C}[n^k]$.

Our proof of Theorem 36 relies on previous results relating randomized learning algorithms and lower bounds. The following connection was established in [44], using ideas from [33, 21]) and most crucially the construction of a downward self-reducible and random self-reducible PSPACE-complete language in [65].

▶ **Theorem 37** (Connection between learning and lower bounds [44, 21, 33]). *There is a* PSPACE*-complete language $L^\star \in \mathsf{DSPACE}(n)$ and a constant $b \in \mathbb{N}$ for which the following holds. Let $\mathfrak{C}$ be any typical circuit class, and $s\colon \mathbb{N} \to \mathbb{N}$ be any function with $s(n) \geq n$. If $\mathfrak{C}[s(n)]$ is learnable to error $\leq n^{-b}$ in time $T(n) \geq n$, then at least one of the following conditions hold:*

(i) $L^\star \notin \mathfrak{C}[s(n)]$.
(ii) $L^\star \in \mathsf{BPTIME}(\mathsf{poly}(T(n)))$.

A self-contained proof of a generalization of Theorem 37 is presented in Section 6. We will also need a consequence of the following diagonalization lemma.

▶ **Lemma 38** (A nonuniform almost everywhere hierarchy for space complexity). *Let $S, S'\colon \mathbb{N} \to \mathbb{N}$ be space-constructible functions such that $S(n) = o(S'(n)), S(n) = \Omega(\log n)$ and $S'(n) = o(2^n)$. There is a language $L \in \mathsf{DSPACE}(S')$ such that $L \notin \mathtt{i.o.}\mathsf{DSPACE}(S)/S$.*

Proof. This is a folklore argument. We define a space-bounded Turing machine M operating in space S' such that $L(M) \notin \mathtt{i.o.}\mathsf{DSPACE}(S)/S$. On inputs of length n, M uses the space-constructibility of S' to compute $S'(n)$ in unary using space $O(S'(n))$. It marks out $S'(n)$ cells on each of its tapes, and if at any point in its computation, it reads an unmarked cell, it halts and rejects. Thus, on any input of length n, M uses space $O(S'(n))$. M also computes and stores $S(n)$ on one of its tapes.

The high-level intuition is that M diagonalizes against machine M_i with advice z, for each $1 \leq i \leq \log n$ and advice $z \in \{0,1\}^{S(n)}$. In particular, for any fixed i and large enough n, M diagonalizes against M_i with any advice $z \in \{0,1\}^{S(n)}$, and hence $L(M)$ satisfies the conclusion of the Lemma.

By a counting argument, there are at most $\log n \cdot 2^{S(n)}$ truth tables of Boolean functions f on n bits such that f is computed by a machine M_i with $1 \leq i \leq \log n$ operating in space $S(n)$ and using $S(n)$ bits of advice. Thus, since $S(n) = o(2^n)$, for large enough n, by the pigeon-hole principle there exists a Boolean function $f'\colon \{0,1\}^n \to \{0,1\}$ which is 0 on all but the first $\log n + S(n)$ inputs of length n, such that f' is not computed by machine M_i with advice z for any i with $1 \leq i \leq \log n$ and $z \in \{0,1\}^{S(n)}$.

M computes such a function iteratively as follows. It processes the inputs of length n in lexicographic order. At stage $i+1$, where $i \geq 0$, M has stored a binary string y_i of length i representing the values of f' on the first i inputs of length n, and M is trying to determine f' on the $(i+1)$-th input of length n. For each machine M_i, $1 \leq i \leq \log n$, and each advice string z for M_i of length $S(n)$, by simulating those M_i's with advice z which do not use space more than $S(n)$ on any of the first i inputs, M determines if the truth table of M_i with advice z is consistent with y_i on the first i inputs. Call such a pair (i, z) a consistent machine-advice pair at stage $i+1$. M sets f' to 0 for the $(i+1)$-th string if a minority of consistent machine-advice pairs halt with 0 on the $(i+1)$-th string, and to 1 otherwise. Determining

whether a minority of consistent machine-advice pairs halt with 0 on the $(i+1)$-th string can be done by merely keeping a count of how many consistent machine-advice pairs halt with 0, and how many halt with 1, which only requires space $O(S(n))$. Note that using the minority value cuts down the number of consistent machine-advice pairs for the next stage by at least a factor of half. This implies that at stage $\log n + S(n)$, there are no consistent machine-advice pairs left, and hence M has successfully diagonalized. It is not hard to see that the overall simulation can be carried out in space $O(S(n))$, using the fact that $S(n) = \Omega(\log n)$. ◀

▶ **Corollary 39** (Diagonalizing in uniform space against non-uniform circuits). *Let $S_1, S_2 : \mathbb{N} \to \mathbb{N}$ be space-constructible functions such that $S_2(n)^2 = o(S_1(n))$, $S_2(n) = \Omega(\log n)$ and $S_1(n) = o(2^n)$. There is a language $L \in \mathsf{DSPACE}(S_1)$ such that $L \notin \mathtt{i.o.}\mathsf{Circuit}[S_2]$. In particular, for each k, there is a language $L_k \in \mathsf{PSPACE}$ such that $L_k \notin \mathsf{Circuit}[n^k]$.*

Proof. Corollary 39 follows from Lemma 38 using the fact that $\mathsf{Circuit}[S] \subseteq \mathsf{DSPACE}(S^2)/S^2$. ◀

In fact, a tighter simulation holds, and therefore a tighter separation in Corollary 39, but we will not need this for our purposes. We are now ready to prove Theorem 36.

Proof of Theorem 36. Let $\mathfrak{C}$ be a typical circuit class. By assumption, $\mathfrak{C}[n^k]$ has a non-trivial learner for each $k > 0$. Since $\mathfrak{C}$ is typical, we can use Lemma 24 to conclude that for each $\varepsilon > 0$ and for each $k > 0$, $\mathfrak{C}[n^k]$ is strongly learnable in time 2^{n^ε}.

Let $L^\star$ be the PSPACE-complete language in the statement of Theorem 37. Using Theorem 37 and the conclusion of the previous paragraph, we have that at least one of the following is true: (1) For all k, $L^\star \notin \mathfrak{C}[n^k]$, or (2) For all $\varepsilon > 0$, $L^\star \in \mathsf{BPTIME}(2^{n^\varepsilon})$.

In case (1), since $L^\star \in \mathsf{DSPACE}(n) \subseteq \mathsf{DTIME}(2^{O(n)})$, we have that for each $k > 0$, $\mathsf{DTIME}(2^{O(n)}) \not\subseteq \mathfrak{C}[n^k]$, and hence also $\mathsf{BPTIME}(2^{O(n)}) \not\subseteq \mathfrak{C}[n^k]$.

In case (2), we have that $L^\star \in \mathsf{BPTIME}(2^{n^\varepsilon})$ for every $\varepsilon > 0$. Since $L^\star$ is PSPACE-complete, this implies that the language L_k in the statement of Corollary 39 is also in $\mathsf{BPTIME}(2^{n^\varepsilon})$, for every fixed $\varepsilon > 0$ and $k \in \mathbb{N}$. (Here the polynomial blowup of instance size in the reduction from L_k to $L^\star$ is taken care of by the universal quantification over ε.) In particular, we have $L_k \in \mathsf{BPTIME}(2^n)$, for every k. Since for any typical circuit class we have $\mathfrak{C}[n^k] \subseteq \mathsf{Circuit}[n^c]$ for a large enough $c = c(k)$, there is a language $L_c \in \mathsf{BPTIME}[2^n]$ such that $L_c \notin \mathfrak{C}[n^k]$. This establishes the desired result. ◀

We mention for completeness that the same approach yields a trade-off involving the running time of the learning algorithm and its accuracy in the hypothesis of Theorem 36.

▶ **Theorem 40** (Trade-off between error and running time). *Let $\mathfrak{C}$ be a typical circuit class, and $\gamma : \mathbb{N} \to (0, 1/2] \cap \mathbb{Q}$ be a polynomial time computable function. If for each k, $\mathfrak{C}[n^k]$ can be learned with advantage at least $\gamma(n)$ in time $\gamma(n)^2 \cdot 2^n / n^{\omega(1)}$, then for each k, $\mathsf{BPTIME}[2^{O(n)}] \not\subseteq \mathfrak{C}[n^k]$.*

Proof (Sketch). The proof is entirely analogous to the argument in Theorem 36. It is enough to observe that such learning algorithms yield the complexity distinguishers required in Lemma 24 via a natural generalization of the proof of Lemma 22. The quantitative trade-off between accuracy and running time is a consequence of Lemma 21. ◀

▶ Remark. Observe that as the advantage $\gamma(n)$ approaches $2^{-n/2}$ from above, the running time required in Theorem 40 becomes meaningless. This quantitative connection between $\gamma(n)$ and the running time is not entirely unexpected. On the one hand, it is a consequence of the concentration bound, which is essentially optimal. But also note that every function

$g\colon \{0,1\}^n \to \{0,1\}$ can be approximated with advantage $\geq 2^{-n/2}$ by a parity function (or its negation), and that heavy fourier coefficients corresponding to such parity functions can be found using membership queries by the Goldreich-Levin Algorithm (see e.g. [53]).

We can expand the scope of application of Theorem 36, using a win-win argument. The more general result below applies to subclasses of Boolean circuits satisfying the very weak requirement that they are closed under projections, rather than just to the more specialized "typical" classes.

▶ **Theorem 41** (Lower bounds from non-trivial learning algorithms for subclasses of circuits)**.** *Let $\mathfrak{C}$ be any subclass of Boolean circuits closed under projections. If for each k, $\mathfrak{C}[n^k]$ has non-trivial learning algorithms, then for each k, $\mathsf{BPTIME}(2^{O(n)}) \not\subseteq \mathfrak{C}[n^k]$.*

Proof. Consider the Circuit Value Problem (CVP), which is complete for $\mathsf{Circuit}[\mathsf{poly}]$ under polynomial size projections. Either CVP is in $\mathfrak{C}[n^c]$ for some fixed c, or it is not. If it is not, then we have the desired lower bound for CVP and hence also for the class $\mathsf{BPTIME}(2^{O(n)})$, which contains this problem. If CVP is in $\mathfrak{C}[n^c]$, then since CVP is closed under poly-size projections, we have by completeness and the assumption of the theorem that for each k, $\mathsf{Circuit}[n^k]$ has non-trivial learning algorithms. Now applying Theorem 36, we have that for each k, $\mathsf{BPTIME}(2^{O(n)}) \not\subseteq \mathsf{Circuit}[n^k]$, which implies that $\mathsf{BPTIME}(2^{O(n)}) \not\subseteq \mathfrak{C}[n^k]$, since $\mathfrak{C}$ is a subclass of Boolean circuits. ◀

▶ Remark. Observe that it is possible to instantiate Theorem 41 for very particular classes such as $\mathsf{AND} \circ \mathsf{OR} \circ \mathsf{THR}$ circuits, and that the lower bound holds for exactly the same circuit class. In particular, there is no circuit depth blow-up.

We get an improved lower bound consequence for the circuit class ACC^0, but under the assumption that subexponential-size circuits are non-trivially learnable. (Recall that there are satisfiability algorithms for such circuits with non-trivial running time [73].)

▶ **Theorem 42** (Improved lower bounds from non-trivial learning algorithms for ACC^0)**.** *If for every depth $d \in \mathbb{N}$ and modulo $m \in \mathbb{N}$ there is $\varepsilon > 0$ such that $\mathsf{ACC}^0_{d,m}[2^{n^\varepsilon}]$ has non-trivial learning algorithms, then $\mathsf{REXP} \not\subseteq \mathsf{ACC}^0[\mathsf{poly}]$.*

Proof. Under the assumption on learnability, using Lemma 24, we have that for each $k > 0$, $\mathsf{ACC}^0[n^k]$ has strong learners running in time $2^{\mathsf{polylog}(n)}$. Now applying Theorem 37, we have that at least one of the following is true for the PSPACE-complete language $L^\star$ in the statement of the theorem: (1) $L^\star \notin \mathsf{ACC}^0[n^k]$ for any k, or (2) $L^\star \in \mathsf{BPQP}$, where BPQP is bounded error probabilistic quasi-polynomial time.

In case (1), we have that $L^\star \notin \mathsf{ACC}^0[\mathsf{poly}]$, and are done as in the proof of Theorem 36.

In case (2), by PSPACE-completeness of $L^\star$, we have that $\mathsf{PSPACE} \subseteq \mathsf{BPQP}$. This implies that $\mathsf{NP} \subseteq \mathsf{BPQP}$, and hence that $\mathsf{NP} \subseteq \mathsf{RQP}$, where RQP is probabilistic quasi-polynomial time with one-sided error. The second implication follows using downward self-reducibility to find a witness for SAT given the assumption that SAT is in BPQP, thus eliminating error on negative instances. Now $\mathsf{NP} \subseteq \mathsf{RQP}$ implies $\mathsf{NEXP} = \mathsf{REXP}$, using a standard translation argument. Williams showed that $\mathsf{NEXP} \not\subseteq \mathsf{ACC}^0[\mathsf{poly}]$, and so it follows that $\mathsf{REXP} \not\subseteq \mathsf{ACC}^0[\mathsf{poly}]$, as desired. ◀

More generally, the same argument combined with the connection between non-trivial satisfiability algorithms and circuit lower bounds [73] imply the following result.

▶ **Corollary 43** (Lower bounds from learning and satisfiability)**.** *Let $\mathfrak{C}$ be any typical circuit class. Assume that for each k, $\mathfrak{C}[n^k]$ admits a non-trivial satisfiability algorithm, and that for some $\varepsilon > 0$, $\mathfrak{C}[2^{n^\varepsilon}]$ admits a non-trivial learning algorithm. Then $\mathsf{REXP} \not\subseteq \mathfrak{C}[\mathsf{poly}]$.*

Recall that randomized learning algorithms and BPP-natural properties are strongly related by results of [19]. We can give still stronger lower bound conclusions from assumptions about P-natural proofs. The idea is to combine the arguments above with an application of the easy witness method of Kabanets [37].

▶ **Theorem 44** (Improved lower bounds from natural proofs). *Let $\mathfrak{C}$ be any subclass of Boolean circuits closed under projections. If there are* P*-natural proofs useful against $\mathfrak{C}[2^{n^\varepsilon}]$ for some $\varepsilon > 0$, then* ZPEXP $\not\subseteq \mathfrak{C}$[poly].

The following immediate consequence is of particular interest.

▶ **Corollary 45** (ACC^0 lower bounds from natural proofs). *If for some $\delta > 0$ there are* P*-natural proofs against $\mathsf{ACC}^0[2^{n^\delta}]$ then* ZPEXP $\not\subseteq \mathsf{ACC}^0$[poly].

In order to prove Theorem 44, we will need the following lemma.

▶ **Lemma 46** (Simulating bounded error with zero error given natural proofs). *Suppose there is a constant $\delta > 0$ such that there are* P*-natural proofs against* Circuit$[2^{n^\delta}]$. *Then* BPEXP = ZPEXP.

Proof. Note that zero-error probabilistic time is trivially contained in bounded-error probabilistic time, so we only need to show that BPEXP $\subseteq$ ZPEXP under the assumption. We will in fact show that BPP $\subseteq$ ZPQP, where ZPQP is zero-error bounded probabilistic quasi-polynomial time. The desired conclusion follows from this using a standard translation argument.

By assumption, there is a natural property $\mathfrak{R}$ useful against Circuit$[2^{n^\delta}]$ for some constant $\delta > 0$, such that $L_{\mathfrak{R}} \in$ P. Let M be any machine operating in bounded-error probabilistic time n^d for some $d > 0$. We define a zero-error machine M' deciding $L(M)$ in quasi-polynomial time as follows. On input x of length n, M' guesses a random string r of size $2^{\mathsf{log}(n)^{d'}}$, where d' is a large enough constant to be defined later. It then checks if $r \in L_{\mathfrak{R}}$ or not, using the polynomial-time decision procedure for the natural property $\mathfrak{R}$. If not, it outputs '?' and halts. If it does, it runs the procedure of Theorem 13 on input n^{2d} in unary and r, to obtain the range of a (polylog$(n), 1/n^{2d}$) PRG against Circuit$[n^{2d}]$. Since $r \in L_{\mathfrak{R}}$, Theorem 13 applies, and the output of the procedure is guaranteed to be the range of such a PRG. M' then runs M on x independently with each element of the range of the PRG used as randomness, and takes the majority vote. This is guaranteed to be correct when $r \in L_{\mathfrak{R}}$, which happens with probability at least $1/2$ by the density property of $\mathfrak{R}$. Thus M' is a zero-error machine, and it is clear that M' can be implemented in quasi-polynomial time. ◀

Proof of Theorem *44*. We proceed as in the proof of Theorem 41. Either CVP is in $\mathfrak{C}$[poly], or it is not. If not, we have the desired lower bound for CVP, and hence for ZPEXP, which contains this problem.

On the other hand, if CVP is in $\mathfrak{C}$[poly], we have that CVP is in $\mathfrak{C}[n^k]$ for some $k > 0$. By the completeness of CVP for poly-size circuits under poly-size projections, and the closure of $\mathfrak{C}$ under projections, we have that Circuit$[n] \subseteq \mathfrak{C}[n^k]$ for some $k > 0$, and hence by a standard translation argument, we have that Circuit$[2^{n^\delta}] \subseteq \mathfrak{C}[2^{n^\varepsilon}]$ for any $\delta < \varepsilon$. By assumption, we have P-natural properties useful against $\mathfrak{C}[2^{n^\varepsilon}]$ and hence we also have P-natural properties useful against Circuit$[2^{n^\delta}]$ for any $\delta < \varepsilon$. Now, applying Lemma 46, we get BPEXP = ZPEXP.

We argue next that under the existence of P-natural properties useful against Circuit$[2^{n^\delta}]$ for a fixed $\delta > 0$, we also have EXPSPACE = BPEXP. The mentioned hypothesis implies that there exist complexity distinguishers against Circuit$[2^{n^\delta}]$ running in deterministic time

$2^{O(n)}$ (the acceptance probability can be amplified using truth-table concatenation). As a consequence, Lemma 23 provides strong learning algorithms for Circuit[poly] running in quasi-polynomial time. By Theorem 37, either PSPACE $\not\subseteq$ Circuit[poly], and we are done, or PSPACE $\subseteq$ BPQP. Now a standard upward translation gives EXPSPACE $\subseteq$ BPEXP, which shows that EXPSPACE = BPEXP.

Altogether, we have EXPSPACE = ZPEXP. Now this collapse and Corollary 39 with $S_1(n) = 2^{\sqrt{n}}$ and $S_2(n) = n^{\log n}$ yield a language $L \in$ ZPEXP such that $L \notin$ Circuit[poly], which completes the proof of Theorem 44. ◀

Recall that the existence of *useful* properties against a circuit class $\mathfrak{C}$ is essentially equivalent to the existence of non-deterministic exponential time lower bounds against $\mathfrak{C}$ [74, 54]. We do not expect a similar equivalence in the case of *natural* properties and lower bounds for probabilistic exponential time. The results described in this section show that natural properties imply such lower bounds. However, if the other direction were true, then any lower for $\mathfrak{C}$ with respect to probabilistic exponential time classes would also provide a non-trivial learning algorithm for $\mathfrak{C}$. In particular, since we believe in separations such as EXP $\not\subseteq$ Circuit[poly], this would imply via the Speedup Lemma that polynomial size circuits can be learned in sub-exponential time, which seems unlikely.

6 Karp-Lipton Collapses for Probabilistic Classes

6.1 A Lemma About Learning with Advice

In this section we will need some notions of computability with advice. While this is a standard notion, we provide some definitions, as bounded-error randomized algorithms taking advice can be defined in different ways.

Recall that an *advice-taking* Turing machine is a Turing machine equipped with an extra tape, the advice tape. At the start of any computation of an advice-taking Turing machine, the input is present on the input tape of the machine and a string called the "advice" on the advice tape of the machine, to both of which the machine has access.

▶ **Definition 47** (Probabilistic time with advice)**.** Let $T\colon \mathbb{N} \to \mathbb{N}$ and $a\colon \mathbb{N} \to \mathbb{N}$ be functions. BPTIME$(T)/a$ is the class of languages $L \subseteq \{0,1\}^*$ for which there is an advice-taking probabilistic Turing machine M which always halts in time $T(n)$ and a sequence $\{z_n\}_{n\in\mathbb{N}}$ of strings such that:

1. For each n, $|z_n| \leq a(n)$.
2. For any input $x \in L$ such that $|x| = n$, M accepts x with probability at least $2/3$ when using advice string z_n.
3. For any input $x \notin L$ such that $|x| = n$, M rejects x with probability at least $2/3$ when using advice string z_n.

Note that in the above definition, there are no guarantees on the behaviour of the machine for advice strings other than the "correct" advice string z_n. In particular, for an arbitrary advice string, the machine does not have to satisfy the bounded-error condition on an input, though it does have to halt within time T.

The notion of resource-bounded computation with advice is fairly general and extends to other models of computation, such as deterministic computation and computation of non-Boolean functions. These extensions are natural, and we will not define them formally.

A slightly less standard notion of computation with advice is learnability with advice. We extend Definition 7 to capture learning with advice by giving the learning algorithm an

advice string, and only requiring the learning algorithm to work correctly for a "correct" advice string of the requisite length.

We will also need the standard notions of downward self-reducibility and random self-reducibility.

▶ **Definition 48** (Downward self-reducibility)**.** A function $f\colon \{0,1\}^* \to \{0,1\}$ is said to be downward self-reducible if there is a polynomial-time oracle procedure $A^f(x)$ such that:

1. On any input x of length n, $A^f(x)$ only makes queries of length $< n$.
2. For every input x, $A^f(x) = f(x)$.

▶ **Definition 49** (Random self-reducibility)**.** A function $f\colon \{0,1\}^* \to \{0,1\}$ is said to be random self-reducible if there are constants $k, \ell \geq 1$ and polynomial-time computable functions $g\colon \{0,1\}^* \to \{0,1\}^*$ and $h\colon \{0,1\}^* \to \{0,1\}$ satisfying the following conditions:

1. For large enough n, for every $x \in \{0,1\}^n$ and for each $i \in \mathbb{N}$ such that $1 \leq i \leq n^k$, $g(i,x,r) \sim U_n$ when $r \sim U_{n^\ell}$.
2. For large enough n and for every function $\tilde{f}_n\colon \{0,1\}^n \to \{0,1\}$ that is $(1/n^k)$-close to f on n-bit strings, for every $x \in \{0,1\}^n$:

   $$f(x) = h(x, r, \tilde{f}_n(g(1,x,r)), \tilde{f}_n(g(2,x,r)), \ldots, \tilde{f}_n(g(n^k,x,r)))$$

   with probability $\geq 1 - 2^{-2n}$ when $r \sim U_{n^\ell}$.

▶ **Theorem 50** (A special PSPACE-complete function [65])**.** *There is a* PSPACE*-complete function $f_{\mathsf{TV}}\colon \{0,1\}^* \to \{0,1\}$ such that f_{TV} is downward self-reducible and random self-reducible.*

Below we consider the learnability of the class of Boolean functions $\{f_{\mathsf{TV}}\}$ that contains only the function f_{TV}.

▶ **Lemma 51** (Learnability with advice for PSPACE implies randomized algorithms)**.** *For any polynomial-time computable non-decreasing function $a\colon \mathbb{N} \to \mathbb{N}$ with $a(n) \leq n$, and for any non-decreasing function $T\colon \mathbb{N} \to \mathbb{N}$ such that $n \leq T(n) \leq 2^n$, if $\{f_{\mathsf{TV}}\}$ is strongly learnable in time T with a bits of advice, then f_{TV} is computable in bounded-error probabilistic time $T(n)^2 \cdot 2^{a(n)} \cdot n^{O(1)}$, and hence* $\mathsf{PSPACE} \subseteq \mathsf{BPTIME}(T(\mathsf{poly}(n))^2 \cdot 2^{a(\mathsf{poly}(n))} \cdot \mathsf{poly}(n))$.

Proof. The argument is based on and extends ideas from [44, 21, 65, 33]. Recall that f_{TV} is the same Boolean function as in the statement of Theorem 50. As stated there, this function is downward self-reducible and random self-reducible. Now suppose $\{f_{\mathsf{TV}}\}$ is learnable in time T with a bits of advice. We design a probabilistic machine M solving f_{TV} on inputs of length n with bounded error in time $T(n)^2 \cdot 2^{a(n)} \cdot n^{O(1)}$. The addition inclusion in the statement of Lemma 51 follows from the completeness of f_{TV}.

Let $x \in \{0,1\}^n$ be the input to M. Let A_{learn} be a $(1/n^{4k}, 1/2^{2n})$-learner for $\{f_{\mathsf{TV}}\}$ that takes $a(n)$ bits of advice and runs in time $T(n)$.[12] Here k is the exponent in the number of queries in the random self-reduction for f_{TV} given by Theorem 50.

[12] In this argument, we do not care about $\mathsf{poly}(n)$ multiplicative factors applied to the final running time, so we can assume the failure probability of the learner to be exponentially small by amplification. This is a standard argument, and we refer to [41] for more details.

Overview. The plan of the proof is that M will use the advice-taking learner to inductively produce, with high probability, circuits computing f_{TV} correctly on inputs of length $1 \dots n$. The crucial aspect is not to allow the size of these circuits to grow too large. There will be n phases in the operation of M – during Phase i, M will produce with high probability a randomized circuit computing f_{TV} on inputs of length i.

Each phase consists of 2 parts. In Part 1 of Phase i, M computes, for each possible advice string z of length $a(i)$ that can be fed to the advice-taking learner on input 1^i, a *candidate deterministic circuit* C_i^z on i-bit inputs of size at most $T(i)$. In order for M to do this, it uses the properties of the learner, as well as the circuits for smaller lengths that have already been computed. The only guarantee on the candidate circuits is that at least one of them is a approximately correct circuit for f_{TV} at length i, in the sense that it is correct on most inputs of this length. In Part 2 of Phase i, M uses the random self-reducibility and downward self-reducibility of f_{TV} to select the "best-performing" candidate among these circuits and compute a "correction" C_i of the best-performing circuit. The circuit C_i will have size $T(i) \cdot \mathsf{poly}(n)$, and with high probability, it will be a randomized circuit that computes f_{TV} correctly on all i-bit inputs, in the sense that on each such string it is correct with overwhelming probability over its internal randomness. At the end of Phase n, M evaluates the circuit C_n on x and outputs the answer.

We now give the details of how Part 1 and Part 2 work for each phase. We will then need to argue that M is correct, and that it is as efficient as claimed. Phase 1, which is the base case for M's inductive operation, is trivial. The circuit C_1 computing f_{TV} correctly on inputs of length 1 is simply hard-coded into M.

Now let $i > 1$ be an integer. We describe how Part 1 and Part 2 of Phase i work, assuming inductively that M already has stored in memory a sequence of circuit $\{C_j\}$, for $1 \leq j \leq i-1$, such that for each such j, C_j has size at most $T(j) \cdot \mathsf{poly}(n)$, and with all but exponentially small probability, computes f_{TV} correctly on each input of length j.

Part 1. M first uses the polynomial-time computability of a to compute $a(i)$. It then cycles over strings $z \in \{0,1\}^{a(i)}$, and for each string z it does the following. It simulates $A_{\mathsf{learn}}(1^i)$ with advice z. Each time A_{learn} makes a membership query of length i, M answers the membership query using the downward self-reducibility of f_{TV} as follows. If the downward self-reduction makes a query of length $j < i$, M answers it by running the stored circuit C_j on the corresponding query.

If $A_{\mathsf{learn}}(1^i)$ with advice z does not halt with an output that is a circuit on i bits, M sets C_i^z to be a trivial circuit on i bits, say the circuit that always outputs 0. Otherwise M sets C_i^z to be the circuit output by the learning algorithm. Since A_{learn} is guaranteed to halt in time $T(i)$ for every advice string, the circuit C_i^z has size at most $T(i)$.

Part 2. M samples strings $y_1, \dots, y_t$, where $t = n^{10k}$, uniformly and independently at random amongst i-bit strings. It computes "guesses" $b_1, \dots, b_t \in \{0,1\}$ for the values of f_{TV} on these inputs by running the downward self-reducibility procedure for f_{TV}, and answering any queries of length $j < i$ using the stored circuits C_j. Then, for each advice string z, it simulates C_i^z on each input y_ℓ, where $1 \leq \ell \leq t$, and computes the fraction ρ_z of inputs y_ℓ for which $C_i^z(y_\ell) = b_\ell$. Let z_{max} be the advice string z for which $\rho_{z_{\mathsf{max}}}$ is maximum among all such advice strings. Let D_i be the (deterministic) circuit $C_i^{z_{\mathsf{max}}}$. M produces a randomized circuit C_i from D_i as follows. C_i applies the random self-reduction procedure for f_{TV} $O(n)$ times independently, using the circuit D_i to answer the random queries to f_{TV}, and outputs the majority answer of these runs. Note that C_i can easily be implemented in size $T(i) \cdot \mathsf{poly}(n)$,

using the fact that the random self-reduction procedure runs in polynomial time. (We stress that C_i is a randomized circuit even though D_i is deterministic.)

It is sufficient to argue that M halts in time $T(n) \cdot \mathsf{poly}(n) \cdot 2^{a(n)}$, and that the final circuit C_n computed by M is a correct randomized circuit for f_{TV} on inputs of length n with high probability over the random choices of M.

Complexity of M. We will show that M uses time at most $T(i)^2 \cdot \mathsf{poly}(n) \cdot 2^{a(i)}$ in Phase i, and computes a circuit C_i of size at most $T(i) \cdot \mathsf{poly}(n)$. Since a and T are non-decreasing, this implies that M uses time at most $T(n)^2 \cdot \mathsf{poly}(n) \cdot 2^{a(n)}$ in total. We will analyze Part 1 and Part 2 separately.

The first step in Part 1, which is computing $a(i)$, can be done in time $\mathsf{poly}(n)$. For each z, simulating the learner and computing the circuit C_i^z can be done in time at most $T(i) \cdot T(i-1) \cdot \mathsf{poly}(n)$, since the learner runs in time $T(i)$ and makes at most that many oracle queries, each of which can be answered by simulating a circuit C_j of size at most $T(j) \cdot \mathsf{poly}(n)$, where $j \leq i-1$. There are $2^{a(i)}$ advice strings z which M cycles over, hence the total time taken by M in Part 1 of Phase i is at most $T(i)^2 \cdot 2^{a(i)} \cdot \mathsf{poly}(n)$ by the non-decreasing property of T.

In Part 2 of Phase i, computing the bits $b_1, \ldots, b_t$ takes time at most $T(i) \cdot \mathsf{poly}(n)$, since the downward self-reducibility procedure runs in time $\mathsf{poly}(n)$, and every query can be answered by simulation of a circuit C_j with $j < i$ in time at most $T(i) \cdot \mathsf{poly}(n)$. For each C_i^z, computing the fraction ρ_z takes time at most $T(i) \cdot \mathsf{poly}(n)$, since it involves simulating C_i^z on $\mathsf{poly}(n)$ inputs, and C_i^z is of size at most $T(i)$. Doing this for each z takes time at most $T(i) \cdot 2^{a(i)} \cdot \mathsf{poly}(n)$ time, as there are $2^{a(i)}$ possible advice strings of length $a(i)$. Computing z_{max} takes time $\mathsf{poly}(n)$, and then computing the "corrected" circuit C_i takes time at most $T(i) \cdot \mathsf{poly}(n)$, since the random self-reducibility procedure runs in polynomial time and can therefore be simulated using polynomial-size circuits.

Correctness of M. Clearly Phase 1 concludes with a correct circuit C_1 for f_{TV} on 1-bit inputs. We will argue inductively that, given that the randomized circuit C_{i-1} computed at the end of Phase $i-1$ is a correct circuit for f_{TV} on inputs of length $i-1$ such that its error probability is at most 2^{-2n} on any input, with all but exponentially small probability over the random choices of M in Phase i, the randomized circuit C_i computed at the end of Phase i is a correct circuit for f_{TV} on inputs of length i, with error probability at most 2^{-2n} on any input. By a union bound over the phases, it follows from this that with all but exponentially small probability, the final circuit C_n is a correct randomized circuit for f_{TV} on inputs of length n (with error probability at most 2^{-2n}), and hence that carrying out all the phases and then simulating C_n on x yields the correct value $f_{\mathsf{TV}}(x)$ with overwhelming probability.

Therefore our task reduces to arguing the correctness of Phase i given the correctness of Phase $i-1$, for an arbitrary i such that $1 < i \leq n$. We discuss the correctness of Part 1 and Part 2 separately.

In Part 1, we argue that with all but exponentially small probability, at least one of the circuits C_i^z computes f_{TV} correctly on all but a $1/i^{3k}$ fraction inputs of length i. Consider the string z_i of length $a(i)$ that is the "correct" advice string for A_{learn} on input 1^i. We only analyze Part 1 for the advice string z_i – the other advice strings are irrelevant to our analysis of correctness for this part. A_{learn} with advice z_i is a correct learner for $\{f_{\mathsf{TV}}\}$; hence with probability at least $1 - 2^{-2i}$, it outputs a circuit that computes f_{TV} on at least a $1 - 1/i^{4k}$ fraction of inputs of length i, when it is given access to a *correct* oracle for f_{TV}. By running the learner $\mathsf{poly}(n)$ times independently and doing standard amplification, the

success probability can be boosted to $1-2^{-2n}$, while keeping the agreement of the hypothesis with f_{TV} at least $1-1/i^{3k}$, and not affecting the efficiency of M by more than a polynomial factor. M might not be able to answer queries to f_{TV} with perfect accuracy, however by the inductive hypothesis that C_j has error at most 2^{-2n} on any specific input for $j < i$, it follows by a union bound that with probability at least $1 - T(i)2^{-2n} \geq 1 - 2^{-n}$ over the internal randomness of M, the simulation of the learner is correct. Hence with probability at least $1-2^{-n}$, M outputs a circuit $C_i^{z_i}$ during Phase i, Part 1, such that $C_i^{z_i}$ agrees with f_{TV} on at least a $1-1/i^{3k}$ fraction of inputs of length i.

Next we analyze Part 2 of Phase i. By a union bound, with probability at least $1 - \mathsf{poly}(n)/2^{2n}$, the "guesses" $b_1, \ldots, b_t \in \{0,1\}$ are all the correct values for f_{TV} on inputs $y_1, \ldots, y_t \in \{0,1\}^i$, where by construction $t = n^{10k}$. By using a standard concentration bound such as Lemma 19, we have that the estimate ρ_{z_i} is at least $1-1/i^{2k}$ with probability at least $1-2^{-4n}$, and that with probability at least $1-2^{-4n}$ any z such that the agreement ρ_z is at least $1-1/i^{2k}$ must be such that C_i^z agrees with f_{TV} on at least a $1-1/i^{3k/2}$ fraction of inputs of length i. Thus with probability at least $1 - \mathsf{poly}(n)/2^{2n}$, we have that $C_i^{z_{\mathsf{max}}}$ has agreement at least $1-1/i^{3k/2}$ with f_{TV} on inputs of length i. By again using a union bound and a standard concentration bound such as Lemma 19, we have that with all but exponentially small probability, the corrected circuit C_i is a randomized circuit which computes f_{TV} correctly on all inputs of length i, making error $< 2^{-2n}$ on any single input. This completes the inductive argument for correctness. ◀

6.2 Karp-Lipton Results for Bounded-Error Exponential Time

▶ **Lemma 52** (Learnability with advice from distinguishability). *Let $f \in \mathsf{EXP}$ be a Boolean function and $a\colon \mathbb{N} \to \mathbb{N}$ be a advice function.*

1. (High-End Generator) *There is a constant $c \geq 1$ such that for any $\varepsilon \in (0,1]$, there is a sequence of functions $\{G_n^{\mathsf{HE}}\}_{n\in\mathbb{N}}$ with $G_n^{\mathsf{HE}}\colon \{0,1\}^{n^c} \to \{0,1\}^{2^{n^\varepsilon}}$ computable in deterministic time $2^{O(n^c)}$ such that if there is a probabilistic procedure $A(1^n)$ taking $a(n)$ bits of advice and running in time $2^{O(n^\varepsilon)}$, and outputting a circuit distinguisher for $G_n^{\mathsf{HE}}(U_{n^c})$ with constant probability for all large enough n, then $\{f\}$ is strongly learnable in time $2^{O(n^\varepsilon)}$ with $a(n)$ bits of advice.*
2. (Low-End Generator) *There is a constant $c \geq 1$ such that for any $d \geq 1$, there is a sequence of functions $\{G_n^{\mathsf{LE}}\}_{n\in\mathbb{N}}$ with $G_n^{\mathsf{LE}}\colon \{0,1\}^{n^c} \to \{0,1\}^{2^{(\log n)^d}}$ computable in deterministic time $2^{O(n^c)}$ such that if there is a probabilistic quasipolynomial-time procedure $A(1^n)$ taking $a(n)$ bits of advice and outputting a circuit distinguisher for $G_n^{\mathsf{LE}}(U_{n^c})$ with constant probability for all large enough n, then $\{f\}$ is strongly learnable in quasi-polynomial time with $a(n)$ bits of advice.*

Proof (Nutshell). This follows from the reconstruction procedure for the Nisan-Wigderson generator together with hardness amplification. We refer to [51] for more details. ◀

▶ **Theorem 53** (Low-end Karp-Lipton Theorem for bounded-error exponential time). *If there is a $k \geq 1$ such that $\mathsf{BPE} \subseteq \mathtt{i.o.Circuit}[n^k]$, then $\mathsf{BPEXP} \subseteq \mathtt{i.o.EXP}/O(\log n)$.*

Proof. We will prove the contrapositive. For each bounded-error probabilistic exponential time machine M, we will define for each rational $\varepsilon > 0$ a deterministic exponential-time machine M_ε taking logarithmic advice which attempts to simulate it. If all of the attempted simulations M_ε fail almost everywhere, we will show that $\mathsf{PSPACE} \subseteq \mathsf{BPSUBEXP}$, and we will then use a translation argument and Corollary 39 to conclude that $\mathsf{BPE} \not\subseteq \mathtt{i.o.Circuit}[n^k]$, thus establishing the contrapositive.

Let M be any bounded-error probabilistic machine running in time 2^{m^j}, where m is the input length, and j is a constant. We assume without loss of generality that $j \geq 1$, and that M has error $< 1/4$ on any input. Let $\varepsilon > 0$ be any rational. We define the deterministic exponential-time machine M_ε taking $O(\log m)$ bits of advice on inputs of length m below. It uses the generators $\{G_n^{\mathsf{HE}}\}$ given by Lemma 52 corresponding to the PSPACE-complete language f_{TV} in the statement of Theorem 50, which is clearly in exponential time.

On input x of length m, M_ε first uses the advice on its tape to determine an integer n such that $2^{2m^j} \leq 2^{n^\varepsilon} < 2^{2(m+1)^j}$. Note that any $n \in \mathbb{N}$ satisfying these conditions is such that $n = \Theta(m^{j/\varepsilon})$. Hence there are $\mathsf{poly}(m)$ possibilities for n, and any of these possibilities can be encoded using $O(\log m)$ bits on the advice tape. Given a number i on the advice tape, M_ε can decode the relevant n by determining the i-th number in increasing order satisfying both inequalities. This can be done in $\mathsf{poly}(m)$ time since we can assume ε is hard-coded into M_ε, and any single inequality verification can be done in $\mathsf{poly}(m)$ time. M_ε then computes $R(y) = G_n^{\mathsf{HE}}(y)$ for every string $y \in \{0,1\}^{n^c}$. It simulates M on x using each string $R(y)$ in turn as the randomness for M, and outputs the majority result of these simulations. It is easy to see that M_ε can be implemented to run in $2^{O(n^c)} = 2^{O(m^{cj/\varepsilon})}$ time, i.e, in time that is exponential on its input length m.

If any of the simulations M_ε succeeds on infinitely many input lengths m, we have that $L(M) \in \mathtt{i.o.}\mathsf{EXP}/O(\log m)$. Suppose, contrariwise, that all of the simulations M_ε fail almost everywhere. We will argue that $f_{\mathsf{TV}} \in \mathsf{BPSUBEXP}$ and hence, by completeness of f_{TV}, $\mathsf{PSPACE} \subseteq \mathsf{BPSUBEXP}$.

For any $x \in \{0,1\}^m$, let C_x be the circuit of size at most 2^{2m^j} defined as follows: the input of C_x is the sequence of random bits r used by M in its computation on x. $C_x(r)$ accepts iff M accepts on x using the sequence r of random bits. By the standard translation of deterministic computations into circuits, C_x can be implemented in size at most 2^{2m^j}, using the fact that M halts in time 2^{m^j}.

Fix any $\varepsilon > 0$. Let n be an arbitrary positive integer, and let $m(n)$ be the unique m such that $2^{2m^j} \leq 2^{n^\varepsilon} < 2^{2(m+1)^j}$ (observe that $h(a) \stackrel{\text{def}}{=} 2^{2a^j}$ is an increasing function, so this m is indeed unique if n is not too small). We claim that for every large enough n, there is an input x of length $m(n)$ such that C_x is a distinguisher for $G_n^{\mathsf{HE}}(U_{n^c})$. Indeed, if not, there are infinitely many n such that for all inputs x of length $m(n)$, C_x is not a distinguisher, but this implies that the simulation M_ε on inputs of length $m(n)$ would succeed infinitely often with advice encoding the input length n. Since for each m, there are only finitely many n such that $m = m(n)$, it follows that the simulation M_ε succeeds on infinitely many input lengths with logarithmic advice. But this contradicts our assumption that the simulation M_ε fails almost everywhere.

Now that our claim is established, we define a deterministic procedure $A(1^n)$ taking $O(n^\varepsilon)$ bits of advice and running in time $2^{O(n^\varepsilon)}$, which for each large enough n produces a circuit distinguisher for G_n^{HE}. The procedure A computes $m(n)$ in polynomial time. Note that $m(n) = O(n^{\varepsilon/j}) = O(n^\varepsilon)$, by our assumption that $j \geq 1$. A then interprets its advice as an string x of length $m(n)$. It computes C_x, which it can do given x in time polynomial in the size of C_x, and outputs C_x. The time taken by A is dominated by the time required to compute C_x, which is $2^{O(n^\varepsilon)}$, and the advice used by A is of size $O(n^\varepsilon)$.

By applying Lemma 52, we get that $\{f_{\mathsf{TV}}\}$ is strongly learnable in time $2^{O(n^\varepsilon)}$ with $O(n^\varepsilon)$ bits of advice. By applying Lemma 51, we get that f_{TV} is computable in bounded-error probabilistic time $2^{O(n^\varepsilon)}$. Note that this is the case for every $\varepsilon > 0$, since our choice of ε was arbitrary. Thus we have $f_{\mathsf{TV}} \in \mathsf{BPSUBEXP}$, and hence by completeness that $\mathsf{PSPACE} \subseteq \mathsf{BPSUBEXP}$. Using a standard upward translation argument and applying

Corollary 39, we get that for every $k > 0$, $\mathsf{BPE} \not\subseteq \mathtt{i.o.Circuit}[n^k]$, which is the desired conclusion. ◀

▶ **Theorem 54** (High-end Karp-Lipton Theorem for bounded-error exponential time)**.** *If* $\mathsf{BPEXP} \subseteq \mathtt{i.o.Circuit}[2^{n/3}]$*, then for each* $\varepsilon > 0$*,* $\mathsf{BPEXP} \subseteq \mathtt{i.o.DTIME}(2^{2^{n^\varepsilon}})/n^\varepsilon$*.*

Proof (Sketch). The proof is entirely analogous to the proof of Theorem 53, except that we use generators G_n^{LE} rather than the generators G_n^{HE}, adjusting other parameters accordingly. We get that either $\mathsf{PSPACE} \subseteq \mathsf{BPQP}$, or that for every $\varepsilon > 0$, $\mathsf{BPEXP} \subseteq \mathtt{i.o.DTIME}(2^{2^{n^\varepsilon}})/n^\varepsilon$. In the first case, by upward translation, we get that $\mathsf{EXPSPACE} = \mathsf{BPEXP}$, and then by using Corollary 39, we conclude that $\mathsf{BPEXP} \not\subseteq \mathtt{i.o.Circuit}[2^{n/3}]$. ◀

▶ **Theorem 55** (Low-end fully uniform Karp-Lipton style theorem for probabilistic time)**.** *If there is a* $k \geq 1$ *such that* $\mathsf{BPE} \subseteq \mathtt{i.o.Circuit}[n^k]$*, then* $\mathsf{REXP} \subseteq \mathtt{i.o.EXP}$*.*

Proof (Sketch). We use the crucial fact that the union of hitting sets is also a hitting set to eliminate the advice in the simulation. The argument is the same as in the proof of Theorem 53, except that the simulating machine M_ε runs M on x using as randomness R every element in turn that is in the range of G_n^{HE} for every n such that $2^{2m^j} \leq 2^{n^\varepsilon} < 2^{2(m+1)^j}$, accepting if and only if any of these runs accepts. Note that M_ε does not take advice. We do not need to give the "correct" n as advice to the machine because, if any n in the interval produces an accepting path (corresponding to a string in the range of the generator), then $\bigcup_n G_n^{\mathsf{HE}}(U_{n^c})$ for n as above contains an accepting path for M on x. Finally, we observe that computing the range of the generator for every such n does not blow-up the complexity of the simulation by more than a polynomial factor. ◀

▶ **Theorem 56** (High-end fully uniform Karp-Lipton style theorem for probabilistic time)**.** *If* $\mathsf{BPEXP} \subseteq \mathtt{i.o.Circuit}[2^{n/3}]$*, then* $\mathsf{REXP} \subseteq \mathtt{i.o.ESUBEXP}$*.*

Proof (Sketch). The proof is entirely analogous to the proof of Theorem 55, except that we use generators G_n^{LE} rather than the generators G_n^{HE}, adjusting other parameters accordingly. ◀

These results can be combined with a Karp-Lipton collapse for deterministic exponential time. For instance, the following holds.

▶ **Corollary 57.** *If there is* $k \in \mathbb{N}$ *such that* $\mathsf{BPE} \subseteq \mathsf{Circuit}[n^k]$*, then* $\mathsf{REXP} \subseteq \mathtt{i.o.MA}$*.*

Proof. It follows from the hypothesis that $\mathsf{E} \subseteq \mathsf{Circuit}[n^k]$, and hence $\mathsf{EXP} \subseteq \mathsf{Circuit}[\mathsf{poly}]$ by translation. This in turn implies that $\mathsf{EXP} = \mathsf{MA}$ [12]. Moreover, the hypothesis gives $\mathsf{REXP} \subseteq \mathtt{i.o.EXP}$ using Theorem 55. Consequently, we get $\mathsf{REXP} \subseteq \mathtt{i.o.MA}$, which completes the proof. ◀

6.3 Karp-Lipton Results for Zero-Error Exponential Time

▶ **Lemma 58** (Fully uniform simulations using easy witness and truth-table concatenation)**.** *Either* $\mathsf{BPP} \subseteq \mathsf{ZPQP}$*, or* $\mathsf{ZPEXP} \subseteq \mathtt{i.o.ESUBEXP}$*.*

Proof. We use the "easy witness" method of Kabanets [37]. Let M be any probabilistic Turing machine with zero error running in time 2^{m^j} for some $j \geq 1$, such that on each random computation path of M on any input x, the output is either the correct answer for M on x or '?', and moreover the probability of outputting '?' is less than 2^{-2m} for any input $x \in \{0,1\}^m$. For each $\varepsilon > 0$, we define the following attempted deterministic simulation M_ε

for M. On input x of length m, M_ε cycles over all circuits C of size $2^{m^{\varepsilon/2}}$ on m^j inputs. For each such circuit, it explicitly computes the truth table $\mathsf{tt}(C)$ of the circuit C, and runs M on x with $\mathsf{tt}(C)$ as randomness. If the run accepts, it accepts; if the run rejects, it rejects. If the run outputs '?', it moves on to the next circuit C in lexicographic order of circuit encodings. If all runs output '?', the machine rejects. It should be clear that the simulation M_ε runs in deterministic time $\leq 2^{2^{m^\varepsilon}}$ on inputs of length m for any sufficiently large m, and only accepts inputs $x \in L(M)$.

If for each $\varepsilon > 0$, we have that the simulation M_ε solves $L(M)$ correctly on all inputs of length m for infinitely many input lengths m, we have that $L(M) \subseteq \mathtt{i.o.}\mathsf{ESUBEXP}$.

Suppose, on the contrary, that there is some $\varepsilon > 0$ such that the simulation M_ε fails on at least one input x of each large enough input length m. We show how to use this to decide every language in BPP in ZPQP.

Let N be any bounded-error probabilistic machine running in time at most n^k for some constant k and large enough n. Assume without loss of generality that N has error $\leq 1/10$ on any input of length n. We use M to give a zero-error simulation N' of N on all inputs of large enough length. Given an input y of length n, N' simulates M on each input x of length $m(n) \stackrel{\text{def}}{=} (\lceil \log n \rceil)^d$ in turn, for some constant $d \geq 1$ to be specified later. If M outputs '?', N' outputs '?', otherwise it moves on to the next input in lexicographic order. If running M gives '?' outputs for every input x of length $m(n)$, N' outputs '?'. Otherwise, N' concatenates the random strings used on the computation paths of M for each input of length $m(n)$ into a single string R_n of length $O(2^{\mathsf{polylog}(n)})$. It then uses R_n as the truth-table of the hard function for the generator in Theorem 13, setting parameters so that at least n^{2k} pseudorandom bits are produced by the generator. It cycles over all possible seeds of the generator and runs N using each output in turn as the sequence of random choices, accepting if and only if a majority of runs accepts.

Setting d to be a large enough constant depending on j, k, ε and the constant c in the statement of Theorem 13, it can be shown that this simulation can be done in quasi-polynomial time, and that it is correct for each input y of large enough length whenever N' does not output '?'. The key here is that by the failure of M_ε for at least one input of any large enough length, the string R_n is guaranteed to be hard enough that the generator is correct. This is because R_n contains a subfunction of sufficiently large worst-case circuit complexity. Hence cycling over all seeds of the generator and taking the majority value gives the correct answer for N on input y. Finally, under our initial assumption that M has exponentially small failure probability, by a union bound the probability that N' outputs '?' on any large enough input is small. This concludes the proof that $\mathsf{BPP} \subseteq \mathsf{ZPQP}$. ◀

▶ **Theorem 59** (High-end fully uniform Karp-Lipton theorem for zero-error exponential time)**.** *If* $\mathsf{ZPEXP} \subseteq \mathtt{i.o.}\mathsf{Circuit}[2^{n/3}]$*, then* $\mathsf{ZPEXP} \subseteq \mathtt{i.o.}\mathsf{ESUBEXP}$*.*

Proof. Observe that the proof of Theorem 56 establishes that if $\mathsf{REXP} \not\subseteq \mathtt{i.o.}\mathsf{ESUBEXP}$, then $\mathsf{PSPACE} \subseteq \mathsf{BPQP}$. By Lemma 58, if $\mathsf{ZPEXP} \not\subseteq \mathtt{i.o.}\mathsf{ESUBEXP}$, then $\mathsf{BPP} \subseteq \mathsf{ZPQP}$, and hence by upward translation, $\mathsf{BPQP} = \mathsf{ZPQP}$. Putting these together, we have that if $\mathsf{ZPEXP} \not\subseteq \mathtt{i.o.}\mathsf{ESUBEXP}$, then $\mathsf{PSPACE} \subseteq \mathsf{ZPQP}$. Now by upward translation, we have that $\mathsf{EXPSPACE} = \mathsf{ZPEXP}$, and hence by Corollary 39, we get $\mathsf{ZPEXP} \not\subseteq \mathtt{i.o.}\mathsf{Circuit}[2^{n/3}]$. ◀

We have learned from Valentine Kabanets (private communication) that he has independently established Theorem 59 in an unpublished manuscript.

In fact, we can get a non-trivial consequence from the *weakest* possible non-trivial assumption about the circuit size of Boolean functions computable in zero-error exponential time. This extension of Theorem 59 relies on the following simple lemma.

▶ **Lemma 60** (Maximally hard functions in exponential space). *Let* $s_{\mathsf{max}}\colon \mathbb{N} \to \mathbb{N}$ *be such that for each* $n \in \mathbb{N}$, $s_{\mathsf{max}}(n)$ *is the maximum circuit complexity among Boolean functions on* n *bits. Then* $\mathsf{EXPSPACE} \not\subseteq \mathtt{i.o.Circuit}[s_{\mathsf{max}} - 1]$.

Proof (Sketch). The proof is by simple diagonalization. In exponential space, we can systematically list the truth tables of Boolean functions on n bits, and maintain the one with the highest circuit complexity. To compute the circuit complexity of a listed truth table can be done by cycling over all circuits, starting from the smallest one, and checking for each circuit whether it computes the given truth table. Once the truth table of a function with maximum circuit complexity has been computed, we simply look up the corresponding entry in the truth table for any particular input. ◀

Now by using the same proof as for Theorem 59 but applying Lemma 60 instead of Corollary 39, we have the following stronger version of Theorem 59. (We note that Theorems 54 and 56 admit similar extensions.)

▶ **Theorem 61** (Strong Karp-Lipton Theorem for zero-error probabilistic exponential time). *Let* $s_{\mathsf{max}}\colon \mathbb{N} \to \mathbb{N}$ *be such that for each* $n \in \mathbb{N}$, $s_{\mathsf{max}}(n)$ *is the maximum circuit complexity among Boolean functions on* n *bits. If* $\mathsf{ZPEXP} \subseteq \mathtt{i.o.Circuit}[s_{\mathsf{max}} - 1]$, *then* $\mathsf{ZPEXP} \subseteq \mathtt{i.o.ESUBEXP}$.

7 Hardness of the Minimum Circuit Size Problem

We will be dealing with various notions of non-uniform reduction to versions of the Minimum Circuit Size Problem (MCSP). Reductions computable in a non-uniform class $\mathfrak{C}$ are formalized using oracle $\mathfrak{C}$-circuits, which are $\mathfrak{C}$-circuits with oracle gates. We only use oracle circuits where oracle gates appear all at the same level. In this setting, we can define size and depth of oracle circuits to be the size and depth respectively of the oracle circuits with oracle gates replaced by AND/OR gates.

▶ **Definition 62** (Non-uniform Reductions). Let $\mathfrak{C}$ be a typical class of circuits, and L and L' be languages.

- (m-reduction) We say L $\mathfrak{C}$-reduces to L' via m-reductions if there is a sequence of poly-size oracle $\mathfrak{C}$-circuits computing the slices L_n of L when the circuits are given oracle L', and such that each oracle circuit has a single oracle gate, which is also the top gate of the circuit.
- (tt-reduction) We say L $\mathfrak{C}$-reduces to L' via tt-reductions if there is a sequence of poly-size oracle $\mathfrak{C}$-circuits computing the slices L_n of L when the circuits are given oracle L', and such that no oracle circuit has a directed path from one oracle gate to another.
- (ε-approximate reductions) We extend these notions to hold between approximations of a language. Given a function $\varepsilon\colon \mathbb{N} \to [0,1]$ and languages L and L', we say that L reduces to ε-approximating L' under a certain notion of reduction if for each $\widetilde{L'}$ which agrees with L' on at least a $1 - \varepsilon(n)$ fraction of inputs of length n for large enough n, L reduces to $\widetilde{L'}$ under that notion of reduction. We say that ε-approximating L reduces to L' if there is a language $\widetilde{L}$ which agrees with L on at least a $1 - \varepsilon(n)$ fraction of inputs of length n for large enough n, such that $\widetilde{L}$ reduces to L'. More generally, we say that ε-approximating L reduces to ε'-approximating L' under a certain notion of reduction if for any language $\widetilde{L'}$ that $\varepsilon'(n)$-approximates L' on inputs of length n for large enough n, there is a language $\widetilde{L}$ that $\varepsilon(n)$-approximates L on inputs of length n for large enough n, and a corresponding reduction from $\widetilde{L}$ to $\widetilde{L'}$.

- (Parameterized reduction) If a reduction is not computed by polynomial size circuits, we extend these definitions in the natural way, and say that L $\mathfrak{C}[s]$-reduces to L', where $s\colon \mathbb{N} \to \mathbb{N}$ is the appropriate circuit size bound.

The following proposition is immediate from the definitions and the fact that typical circuit classes are closed under composition.

▶ **Proposition 63** (Transitivity of reductions). *Let $\mathfrak{C}$ be a typical circuit class, L, L', L'' be languages, and $\varepsilon, \varepsilon', \varepsilon''\colon \mathbb{N} \to [0,1]$ be functions.*

(i) *If L $\mathfrak{C}$-reduces to L' via m-reductions (resp. tt-reductions) and L' $\mathfrak{C}$-reduces to L'' via m-reductions (resp. tt-reductions), then L $\mathfrak{C}$-reduces to L'' via m-reductions (resp. tt-reductions).*

(ii) *If $\varepsilon(\mathsf{poly}(n))$-approximating L $\mathfrak{C}$-reduces to $\varepsilon'(\mathsf{poly}(n))$-approximating L' via m-reductions (tt-reductions) and $\varepsilon'(\mathsf{poly}(n))$-approximating L' $\mathfrak{C}$-reduces to $\varepsilon''(\mathsf{poly}(n))$-approximating L'' via m-reductions (tt-reductions), it follows that $\varepsilon(\mathsf{poly}(n))$-approximating L $\mathfrak{C}$-reduces to $\varepsilon''(\mathsf{poly}(n))$-approximating L'' via m-reductions (tt-reductions).*

Using the notation introduced above, the following fact is trivial to establish.

▶ **Proposition 64** (Relation between parameterized and unparameterized versions of MCSP). *For any typical circuit class $\mathfrak{C}$, $\mathsf{MCSP}\text{-}\mathfrak{C}[2^{n/2}]$ AC^0-reduces to $\mathsf{MCSP}\text{-}\mathfrak{C}$ via m-reductions.*

▶ **Theorem 65** (Hardness of MCSP for weakly approximating functions in typical circuit classes). *Let $\mathfrak{C}$ be a typical circuit class that contains $\mathsf{AC}^0[p]$, for some fixed prime p. For every Boolean function $f \in \mathfrak{C}[n^k]$ there exists $c = c(k,\delta) \in \mathbb{N}$ such that $(1/2 - \Omega(1/n^c))$-approximating f AC^0-reduces to $\mathsf{MCSP}\text{-}\mathfrak{C}[2^{n/2}]$ via tt-reductions, as well as to any property with density at least $1/4$ that is useful against $\mathfrak{C}[2^{\delta n}]$ for some fixed $\delta \in (0,1)$.*

Proof (Sketch). Let $f = \{f_n\}_{n\in\mathbb{N}}$ be a function in $\mathfrak{C}[n^k]$, where $f_n\colon \{0,1\}^n \to \{0,1\}$ and $\mathsf{AC}^0[p] \subseteq \mathfrak{C}[\mathsf{poly}]$. Further, let $0 < \delta < 1$ be a constant. We let

$$\mathsf{NW}_c(f_n) \stackrel{\text{def}}{=} \{g_z\colon \{0,1\}^{c\log n} \to \{0,1\} \mid z \in \{0,1\}^{\Theta(n^2)}\}$$

be the family (multiset) of functions obtained by instantiating the Nisan-Wigderson [51] construction with the $\mathsf{AC}^0[p]$-computable designs from [19] and f_n. A bit more precisely, each g_z is a function specified by a seed z of length $\Theta(n^2)$, the family of sets $\mathcal{S}_n = \{S_w \subseteq [\Theta(n^2)] \mid w \in \{0,1\}^{c\log n}\}$, and f_n, and we have $g_z(w) \stackrel{\text{def}}{=} f_n(z_{S_w})$. Here each $S_w \subseteq [\Theta(n^2)]$ contains exactly n elements (S_w is the w-th set in the design), and $z_{S_w} \in \{0,1\}^n$ is the projection of z to coordinates S_w. By taking $c = c(\delta, k)$ sufficiently large and using that $\delta > 0$, $f_n \in \mathfrak{C}[n^k]$, and that the design can be implemented in $\mathfrak{C}[\mathsf{poly}]$, it follows from [51, 19] that for large enough n:

(A) Each g_z is a function on $m \stackrel{\text{def}}{=} c\log n$ input bits of $\mathfrak{C}$-circuit complexity $\leq 2^{\delta m}$.

On the other hand, if $h_m \sim \mathcal{F}_m$ is a uniformly random Boolean function on m input bits, using that $\delta < 1$ it easily follows from a counting argument (e.g. Lemma 20) that for large enough n (recall that $m = c\log n$):

(B) h_m has $\mathfrak{C}$-circuit complexity $> 2^{\delta m}$ with probability $1 - o(1)$.

Consequently, from (A) and (B) we get that an oracle to $\mathsf{MCSP}\text{-}\mathfrak{C}[2^{n/2}]$ (corresponding to $\delta = 1/2$) can be used to distinguish the multiset $\mathsf{NW}_c(f_n)$ (sampled according to $z \sim \{0,1\}^{\Theta(n^2)}$) from a random function on m input bits.

We argue next that it follows from the description of the Nisan-Wigderson reconstruction procedure [51] that there is a *tt*-reduction from $(1/2 - \Omega(1/n^c))$-approximating f_n to MCSP-$\mathfrak{C}[2^{n/2}]$ that is computable by AC^0-circuits. That some non-uniform approximate reduction with oracle access to f_n exists immediately follows from the proof of their main result. That it can be computed in $\mathsf{AC}^0[\mathsf{poly}(n)]$ with oracle access to the distinguisher MCSP-$\mathfrak{C}[2^{n/2}]$ (and without oracle access to f_n) follows by our choice of parameters (in particular, $|S_{w_1} \cap S_{w_2}| = O(\log n)$ for every pair $w_1 \neq w_2$), non-uniformity of the reduction, and the fact that the output of f_n on any particular n-bit input can be hardwired into the (non-uniform) AC^0 circuit computing the reduction. Finally, we remark that the $\Omega(1/n^c)$ advantage in the approximation comes from the truth-table size of each g_z and the hybrid argument in [51], and that we get a *tt*-reduction because the reconstruction procedure is non-adaptive.

In fact, the same argument shows that $(1/2 - \Omega(1/n^c))$-approximating f AC^0-reduces via *tt*-reductions to any property useful against $\mathfrak{C}[2^{\delta n}]$ for some $\delta \in (0,1)$ and with density at least $1/4$, since this suffices to implement the Nisan-Wigderson reconstruction routine. This completes the proof of Theorem 65. ◀

▶ **Corollary 66** (Hardness of the standard circuit version of MCSP). *For any Boolean function $f \in \mathsf{Circuit}[\mathsf{poly}(n)]$, there exists $c \geq 1$ such that $(1/2 - 1/n^c)$-approximating f AC^0-reduces via tt-reductions to* MCSP-Circuit *and to any property with density at least $1/4$ that is useful against* $\mathsf{Circuit}[2^{n/2}]$.

Proof. The second item follows immediately from Theorem 65, since Circuit is typical. The first item follows from Theorem 65, Proposition 64 and Proposition 63. ◀

▶ **Proposition 67** (Hardness amplification for Formula). *There exists a Boolean function $f \in \mathsf{Formula}$ that is* Formula*-hard under AC^0-reductions such that for every integer $d \geq 1$, f TC^0-reduces via tt-reductions to $(1/2 - 1/n^d)$-approximating f.*

Proof (Sketch). This is achieved using a standard hardness amplification argument using the existence of a random self-reducible complete problem in NC^1, as well as the XOR lemma. It is known that the circuits used in the hardness amplification reconstruction procedure and for random-self-reducibility can be implemented in non-uniform TC^0. For more details, we refer to [63, 2, 24]. ◀

▶ **Corollary 68** (Hardness of MCSP for NC^1). *For every Boolean function $f \in \mathsf{Formula}$, f TC^0-reduces to the following problems via tt-reductions*:

1. MCSP-$\mathsf{Formula}[2^{n/2}]$.
2. *Any property useful against* $\mathsf{Formula}[2^{\delta n}]$ *for $\delta \in (0,1)$ and with density at least $1/4$.*
3. MCSP-Formula.
4. MCSP-$\mathfrak{C}$ *for any typical circuit class $\mathfrak{C} \supseteq$* Formula.

Proof. Items 1 and 2 follow from Theorem 65 applied to the typical class Formula, together with Propositions 63 and 67. Item 3 follows from Item 1 and Propositions 63 and 64. Finally, in order to prove Item 4, note that MCSP-$\mathfrak{C}[2^{n/2}]$ is useful against $\mathsf{Formula}[2^{n/2}]$, using the assumption that formulas are subclasses of $\mathfrak{C}$ circuits. Moreover, MCSP-$\mathfrak{C}[2^{n/2}]$ as a property has density $1 - o(1)$, since a random function has circuit complexity higher than $2^{n/2}$ with probability exponentially close to 1 by the usual counting argument. Thus, it follows using the same argument as for Item 2 that MCSP-$\mathfrak{C}[2^{n/2}]$ is TC^0-hard under *tt*-reductions for Formula. Item 4 now follows from this via Propositions 63 and 64. ◀

Hardness results as in Corollary 68 also follow for other classes such as non-uniform logarithmic space and the class of problems reducible to the determinant using non-uniform TC^0 reductions, since these classes also have random self-reducible complete problems and admit worst-case to average-case reducibility in low complexity classes. We will not further elaborate on this here.

A closely related problem is whether a string has high KT complexity (cf. [3]). KT complexity is a version of Kolmogorov complexity, where a string has low complexity if it has a short description from which its bits are efficiently computable. We will not explore consequences for this notion in this work, but we expect that some of our results can be transferred to the problem of whether a string has high KT-complexity using standard observations about the relationship between this problem and MCSP.

8 Open Problems and Further Research Directions

We describe here a few directions and problems that we find particularly interesting, and that deserve further investigation.

Speedups in Computational Learning Theory. One of our main conceptual contributions is the discovery of a surprising speedup phenomenon in learning under the uniform distribution using membership queries (Lemma 24). Naturally, it would be relevant to understand which learning models admit similar speedups. In particular, is there an analogous result for learning under the uniform distribution using random examples? An orthogonal question is to weaken the assumptions on concept classes for which learning speedups hold.

Applications in Machine Learning. Is it possible to use part of the machinery behind the proof of the Speedup Lemma (Lemma 24) to obtain faster algorithms in practice? Notice that speedups are available for classes containing a constant number of layers of threshold gates, as TC^0 is a typical circuit class according to our definition. Since these circuits can be seen as discrete analogues of neural networks, which have proven quite successful in several contexts of practical relevance, we believe that it is worth exploring these implications.

Non-Uniform Circuit Lower Bounds from Learning Algorithms. As discussed in [72], strong lower bounds are open even for seemingly weak classes such as $\mathsf{MOD}_2 \circ \mathsf{AND} \circ \mathsf{THR}$ and $\mathsf{AND} \circ \mathsf{OR} \circ \mathsf{MAJ}$ circuits. We would like to know if the learning approach to non-uniform lower bounds (Theorem 41) can lead to new lower bounds against such heavily constrained circuits. More ambitiously, it would be extremely interesting to understand the learnability of ACC^0, given that the existence of a nontrivial algorithm for large enough circuits implies $\mathsf{REXP} \not\subseteq \mathsf{ACC}^0$ (Theorem 42).

The Frontier of Natural Proofs. Is there a natural property against ACC^0? Williams [73] designed a non-trivial satisfiability algorithm for sub-exponential size ACC^0 circuits, which implies in particular that $\mathsf{NEXP} \not\subseteq \mathsf{ACC}^0$. On the other hand, Corollary 45 shows that the existence of a natural property against such circuits implies the stronger lower bound $\mathsf{ZPEXP} \not\subseteq \mathsf{ACC}^0$.

Connections between Learning, Proofs, Satisfiability, and Derandomization. Together with previous work (e.g. [70, 73, 62, 35]), it follows that non-trivial learning, non-trivial proofs of tautologies (in particular, nontrivial satisfiability algorithms), and non-trivial

derandomization algorithms all imply (randomized or nondeterministic) exponential time circuit lower bounds. These are distinct algorithmic frameworks, and the argument in each case is based on a different set of techniques. Is there a more general theory that is able to explain and to strengthen these connections? We view Corollary 43 as a very preliminary result indicating that a more general theory along these lines might be possible.

Unconditional Nontrivial Zero-Error Simulation of REXP. Establish unconditionally that $\mathsf{REXP} \subseteq \mathtt{i.o.ZPESUBEXP}$. We view this result as an important step towards the ambitious goal of unconditionally derandomizing probabilistic computations, and suspect that it might be within the reach of current techniques. In particular, this would follow if one can improve Lemma 58, which unconditionally establishes that either $\mathsf{BPP} \subseteq \mathsf{ZPQP}$ or $\mathsf{ZPEXP} \subseteq \mathtt{i.o.ESUBEXP}$, to a result of the same form but with REXP in place of ZPEXP.

Learning Algorithms vs. Pseudorandom Functions. The results from Section 4 establish an equivalence between learning algorithms and the lack of pseudorandom functions in a typical circuit class, in the non-uniform exponential time regime. It would be interesting to further investigate this dichotomy, and to understand whether a more uniform equivalence can be established.

Hardness of the Minimum Circuit Size Problem. Show that $\mathsf{MCSP} \notin \mathsf{AC}^0[p]$. We have established that if $\mathsf{MCSP} \in \mathsf{TC}^0$ then $\mathsf{NC}^1 \subseteq \mathsf{TC}^0$. Prove that if $\mathsf{MCSP} \in \mathsf{TC}^0$ then $\mathsf{Circuit[poly]} \subseteq \mathsf{TC}^0$.

Acknowledgments. We thank Eric Allender, Marco Carmosino, Russell Impagliazzo, Valentine Kabanets, Antonina Kolokolova, Jan Krajíček, Tal Malkin, Ján Pich, Rocco Servedio and Ryan Williams for discussions. We also thank Ruiwen Chen for several conversations during early stages of this work.

References

1 Manindra Agrawal and V. Vinay. Arithmetic circuits: A chasm at depth four. In *Symposium on Foundations of Computer Science* (FOCS), pages 67–75, 2008. doi:10.1109/FOCS.2008.32.

2 Eric Allender, Vikraman Arvind, and Fengming Wang. Uniform derandomization from pathetic lower bounds. In *International Workshop on Randomization and Computation* (RANDOM), pages 380–393, 2010. doi:10.1007/978-3-642-15369-3_29.

3 Eric Allender, Harry Buhrman, Michal Koucký, Dieter van Melkebeek, and Detlef Ronneburger. Power from random strings. *SIAM J. Comput.*, 35(6):1467–1493, 2006. doi:10.1137/050628994.

4 Eric Allender and Bireswar Das. Zero knowledge and circuit minimization. In *Symposium on Mathematical Foundations of Computer Science* (MFCS), pages 25–32, 2014. doi:10.1007/978-3-662-44465-8_3.

5 Eric Allender, Joshua A. Grochow, and Cristopher Moore. Graph isomorphism and circuit size. *CoRR*, abs/1511.08189, 2015. URL: http://arxiv.org/abs/1511.08189.

6 Eric Allender, Lisa Hellerstein, Paul McCabe, Toniann Pitassi, and Michael E. Saks. Minimizing disjunctive normal form formulas and AC^0 circuits given a truth table. *SIAM J. Comput.*, 38(1):63–84, 2008. doi:10.1137/060664537.

7 Eric Allender, Dhiraj Holden, and Valentine Kabanets. The minimum oracle circuit size problem. In *Symposium on Theoretical Aspects of Computer Science* (STACS), volume 30 of *LIPIcs*, pages 21–33, 2015. doi:10.4230/LIPIcs.STACS.2015.21.

8 Eric Allender and Michal Koucký. Amplifying lower bounds by means of self-reducibility. *J. ACM*, 57(3), 2010. doi:10.1145/1706591.1706594.

9 Ingo Althöfer. On sparse approximations to randomized strategies and convex combinations. *Linear Algebra Appl.*, 199:339–355, 1994. doi:10.1016/0024-3795(94)90357-3.

10 Dana Angluin. Queries and concept learning. *Machine Learning*, 2(4):319–342, 1987. doi:10.1007/BF00116828.

11 Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Cryptography in NC^0. *SIAM J. Comput.*, 36(4):845–888, 2006. doi:10.1137/S0097539705446950.

12 László Babai, Lance Fortnow, Noam Nisan, and Avi Wigderson. BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3:307–318, 1993. doi:10.1007/BF01275486.

13 László Babai and Shlomo Moran. Arthur-Merlin games: A randomized proof system, and a hierarchy of complexity classes. *J. Comput. Syst. Sci.*, 36(2):254–276, 1988. doi:10.1016/0022-0000(88)90028-1.

14 Avrim Blum, Merrick L. Furst, Michael J. Kearns, and Richard J. Lipton. Cryptographic primitives based on hard learning problems. In *Advances in Cryptology* (CRYPTO), pages 278–291, 1993. doi:10.1007/3-540-48329-2_24.

15 Manuel Blum. A machine-independent theory of the complexity of recursive functions. *J. ACM*, 14(2):322–336, 1967. doi:10.1145/321386.321395.

16 Dan Boneh and Richard J. Lipton. Amplification of weak learning under the uniform distribution. In *Conference on Computational Learning Theory* (COLT), pages 347–351, 1993. doi:10.1145/168304.168372.

17 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.

18 Harry Buhrman and Steven Homer. Superpolynomial circuits, almost sparse oracles and the exponential hierarchy. In *Conference on Foundations of Software Technology and Theoretical Computer Science* (FSTTCS), pages 116–127, 1992. doi:10.1007/3-540-56287-7_99.

19 Marco L. Carmosino, Russell Impagliazzo, Valentine Kabanets, and Antonina Kolokolova. Learning algorithms from natural proofs. In *Conference on Computational Complexity* (CCC), volume 50 of *LIPIcs*, pages 10:1–10:24, 2016. doi:10.4230/LIPIcs.CCC.2016.10.

20 Ruiwen Chen, Valentine Kabanets, Antonina Kolokolova, Ronen Shaltiel, and David Zuckerman. Mining circuit lower bound proofs for meta-algorithms. *Computational Complexity*, 24(2):333–392, 2015. doi:10.1007/s00037-015-0100-0.

21 Lance Fortnow and Adam R. Klivans. Efficient learning algorithms yield circuit lower bounds. *J. Comput. Syst. Sci.*, 75(1):27–36, 2009. doi:10.1016/j.jcss.2008.07.006.

22 Lance Fortnow and Rahul Santhanam. Infeasibility of instance compression and succinct PCPs for NP. *J. Comput. Syst. Sci.*, 77(1):91–106, 2011. doi:10.1016/j.jcss.2010.06.007.

23 Oded Goldreich. *The Foundations of Cryptography – Volume 1, Basic Techniques*. Cambridge University Press, 2001.

24 Oded Goldreich, Noam Nisan, and Avi Wigderson. On Yao's XOR-Lemma. In *Studies in Complexity and Cryptography*, pages 273–301. Springer, 2011. doi:10.1007/978-3-642-22670-0_23.

25 Shafi Goldwasser and Silvio Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In *Symposium on Theory of Computing* (STOC), pages 365–377, 1982. doi:10.1145/800070.802212.

26 Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984. doi:10.1016/0022-0000(84)90070-9.

27 Ryan C. Harkins and John M. Hitchcock. Exact learning algorithms, betting games, and circuit lower bounds. *Transactions on Computation Theory* (TOCT), 5(4):18, 2013. doi:10.1145/2539126.2539130.

28 Shuichi Hirahara and Osamu Watanabe. Limits of minimum circuit size problem as oracle. In *Conference on Computational Complexity* (CCC), volume 50 of *LIPIcs*, pages 18:1–18:20, 2016. doi:10.4230/LIPIcs.CCC.2016.18.

29 John M. Hitchcock and Aduri Pavan. On the NP-completeness of the minimum circuit size problem. In *Conference on Foundation of Software Technology and Theoretical Computer Science* (FSTTCS), volume 45 of *LIPIcs*, pages 236–245, 2015. doi:10.4230/LIPIcs.FSTTCS.2015.236.

30 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.

31 Russell Impagliazzo and Leonid A. Levin. No better ways to generate hard NP instances than picking uniformly at random. In *Symposium on Foundations of Computer Science* (FOCS), pages 812–821, 1990. doi:10.1109/FSCS.1990.89604.

32 Russell Impagliazzo and Avi Wigderson. P = BPP if E requires exponential circuits: Derandomizing the XOR lemma. In *Symposium on Theory of Computing* (STOC), pages 220–229, 1997. doi:10.1145/258533.258590.

33 Russell Impagliazzo and Avi Wigderson. Randomness vs time: Derandomization under a uniform assumption. *J. Comput. Syst. Sci.*, 63(4):672–688, 2001. doi:10.1006/jcss.2001.1780.

34 Jeffrey C. Jackson. An efficient membership-query algorithm for learning DNF with respect to the uniform distribution. *J. Comput. Syst. Sci.*, 55(3):414–440, 1997. doi:10.1006/jcss.1997.1533.

35 Hamid Jahanjou, Eric Miles, and Emanuele Viola. Local reductions. In *International Colloquium on Automata, Languages, and Programming* (ICALP), pages 749–760, 2015. doi:10.1007/978-3-662-47672-7_61.

36 Svante Janson, Tomasz Łuczak, and Andrzej Ruciński. *Random graphs*. Wiley-Interscience, New York, 2000.

37 Valentine Kabanets. Easiness assumptions and hardness tests: Trading time for zero error. *J. Comput. Syst. Sci.*, 63(2):236–252, 2001. doi:10.1006/jcss.2001.1763.

38 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.

39 Richard M. Karp and Richard J. Lipton. Some connections between nonuniform and uniform complexity classes. In *Symposium on Theory of Computing* (STOC), pages 302–309, 1980. doi:10.1145/800141.804678.

40 Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography*. Chapman and Hall/CRC Press, 2007.

41 Michael Kearns and Umesh Vazirani. *An Introduction to Computational Learning Theory*. MIT Press, 1994.

42 Michael J. Kearns and Leslie G. Valiant. Cryptographic limitations on learning boolean formulae and finite automata. *J. ACM*, 41(1):67–95, 1994. doi:10.1145/174644.174647.

43 Subhash Khot and Rishi Saket. Hardness of minimizing and learning DNF expressions. In *Symposium on Foundations of Computer Science* (FOCS), pages 231–240, 2008. doi:10.1109/FOCS.2008.37.

44 Adam Klivans, Pravesh Kothari, and Igor C. Oliveira. Constructing hard functions using learning algorithms. In *Conference on Computational Complexity* (CCC), pages 86–97, 2013. doi:10.1109/CCC.2013.18.

45 Jan Krajícek. *Forcing with Random Variables and Proof Complexity*, volume 382 of *London Mathematical Society Lecture Note Series*. Cambridge University Press, 2011. URL: http://www.cambridge.org/de/academic/subjects/mathematics/logic-categories-and-sets/forcing-random-variables-and-proof-complexity?format=PB.

46 Nathan Linial, Yishay Mansour, and Noam Nisan. Constant depth circuits, fourier transform, and learnability. *J. ACM*, 40(3):607–620, 1993. doi:10.1145/174130.174138.

47 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.

48 William J. Masek. Some NP-complete set covering problems. *Unpublished Manuscript*, 1979.

49 Cody D. Murray and Richard Ryan Williams. On the (non) NP-hardness of computing circuit complexity. In *Conference on Computational Complexity* (CCC), volume 33 of *LIPIcs*, pages 365–380, 2015. doi:10.4230/LIPIcs.CCC.2015.365.

50 Moni Naor and Omer Reingold. Number-theoretic constructions of efficient pseudo-random functions. *J. ACM*, 51(2):231–262, 2004. doi:10.1145/972639.972643.

51 Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994. doi:10.1016/S0022-0000(05)80043-1.

52 Noam Nisan and David Zuckerman. Randomness is linear in space. *J. Comput. Syst. Sci.*, 52(1):43–52, 1996. doi:10.1006/jcss.1996.0004.

53 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.

54 Igor C. Oliveira. *Unconditional Lower Bounds in Complexity Theory*. PhD thesis, Columbia University, 2015.

55 Christos H. Papadimitriou. *Computational complexity*. Addison-Wesley, 1994.

56 Ramamohan Paturi and Pavel Pudlák. On the complexity of circuit satisfiability. In *Symposium on Theory of Computing* (STOC), pages 241–250, 2010. doi:10.1145/1806689.1806724.

57 Ján Pich. Circuit lower bounds in bounded arithmetics. *Ann. Pure Appl. Logic*, 166(1):29–45, 2015. doi:10.1016/j.apal.2014.08.004.

58 Alexander A. Razborov. Pseudorandom generators hard for k-DNF resolution and polynomial calculus resolution. *Ann. of Math.* (2), 181(2):415–472, 2015. doi:10.4007/annals.2015.181.2.1.

59 Alexander A. Razborov and Steven Rudich. Natural proofs. *J. Comput. Syst. Sci.*, 55(1):24–35, 1997. doi:10.1006/jcss.1997.1494.

60 Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6), 2009. doi:10.1145/1568318.1568324.

61 Rahul Santhanam. Circuit lower bounds for Merlin–Arthur classes. *SIAM J. Comput.*, 39(3):1038–1061, 2009. doi:10.1137/070702680.

62 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.

63 Ronen Shaltiel and Emanuele Viola. Hardness amplification proofs require majority. *SIAM J. Comput.*, 39(7):3122–3154, 2010. doi:10.1137/080735096.

64 Srikanth Srinivasan. A compression algorithm for $AC^0[\oplus]$ circuits using certifying polynomials. *Electronic Colloquium on Computational Complexity* (ECCC), 22:142, 2015. URL: https://eccc.weizmann.ac.il/report/2015/142/.

65 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.

66 John v. Neumann. Zur Theorie der Gesellschaftsspiele. *Math. Ann.*, 100(1):295–320, 1928. doi:10.1007/BF01448847.

67 Salil P. Vadhan and Colin J. Zheng. A uniform min-max theorem with applications in cryptography. In *Cryptology Conference* (CRYPTO), pages 93–110, 2013. doi:10.1007/978-3-642-40041-4_6.

68 Leslie Valiant. A theory of the learnable. *Communications of the ACM*, pages 1134–1142, 1984.

69 Ilya Volkovich. On learning, lower bounds and (un)keeping promises. In *International Colloquium on Automata, Languages, and Programming* (ICALP), pages 1027–1038, 2014. doi:10.1007/978-3-662-43948-7_85.

70 Ryan Williams. Improving exhaustive search implies superpolynomial lower bounds. *SIAM J. Comput.*, 42(3):1218–1244, 2013. doi:10.1137/10080703X.

71 Ryan Williams. Algorithms for circuits and circuits for algorithms: Connecting the tractable and intractable. *Proceedings of the International Congress of Mathematicians* (ICM), Volume IV:659–682, 2014.

72 Ryan Williams. New algorithms and lower bounds for circuits with linear threshold gates. In *Symposium on Theory of Computing* (STOC), pages 194–202, 2014. doi:10.1145/2591796.2591858.

73 Ryan Williams. Nonuniform ACC circuit lower bounds. *J. ACM*, 61(1):2:1–2:32, 2014. doi:10.1145/2559903.

74 Ryan Williams. Natural proofs versus derandomization. *SIAM J. Comput.*, 45(2):497–529, 2016. doi:10.1137/130938219.

75 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 Quadratic Lower Bound for Homogeneous Algebraic Branching Programs*

Mrinal Kumar

Rutgers University, Newark, NJ, USA
mrinal.kumar@rutgers.edu

Abstract

An algebraic branching program (ABP) is a directed acyclic graph, with a start vertex s, and end vertex t and each edge having a weight which is an affine form in variables $x_1, x_2, \ldots, x_n$ over an underlying field. An ABP computes a polynomial in a natural way, as the sum of weights of all paths from s to t, where the weight of a path is the product of the weights of the edges in the path. An ABP is said to be homogeneous if the polynomial computed at every vertex is homogeneous. In this paper, we show that any homogeneous algebraic branching program which computes the polynomial $x_1^n + x_2^n + \ldots + x_n^n$ has at least $\Omega(n^2)$ vertices (and edges).

To the best of our knowledge, this seems to be the first non-trivial super-linear lower bound on the number of vertices for a general *homogeneous* ABP and slightly improves the known lower bound of $\Omega(n \log n)$ on the number of edges in a general (possibly *non-homogeneous*) ABP, which follows from the classical results of Strassen (1973) and Baur & Strassen (1983).

On the way, we also get an alternate and unified proof of an $\Omega(n \log n)$ lower bound on the size of a *homogeneous* arithmetic circuit (follows from [Strassen, 1973] and [Baur & Strassen, 1983]), and an $n/2$ lower bound (n over reals) on the determinantal complexity of an explicit polynomial [Mignon & Ressayre, 2004], [Cai, Chen & Li, 2010], [Yabe, 2015]. These are currently the best lower bounds known for these problems for any explicit polynomial, and were originally proved nearly two decades apart using seemingly different proof techniques.

1998 ACM Subject Classification I.1.1 Expressions and Their Representation

Keywords and phrases algebraic branching programs, arithmetic circuits, determinantal complexity, lower bounds

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.19

1 Introduction

The question of proving superpolynomial lower bounds on the size of arithmetic circuits for an explicit polynomial family is a fundamental problem in the area of algebraic complexity theory. Unfortunately, the state of art for this problem is quite unsatisfying and the best lower bound known for general arithmetic circuits is an $\Omega(n \log d)$ lower bound for the polynomial $P_{(n,d)} = \sum_{i=1}^{n} x_i^d$, proved by Strassen [16] and Baur & Strassen [2] more than three decades ago. The absence of substantial progress on the general question has led to focus on the question of proving better lower bounds for interesting restricted classes of arithmetic circuits. Arithmetic formula, non-commutative circuits, bounded depth circuits, multilinear formulas and monotone arithmetic circuits are some restricted classes of arithmetic circuits which have been studied from this point of view, and for many of these classes substantial progress has been made on the question of proving lower bounds. We refer the reader to the surveys

* Research supported in part by a Simons Graduate Fellowship.

32nd Computational Complexity Conference (CCC 2017).
Editor: Ryan O'Donnell; Article No. 19; pp. 19:1–19:16

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

of Shpilka-Yehudayoff [13] and Saptharishi [11] and the references therein for an overview of these results. One such restricted model of computation, which will be the primary focus of this paper is the model of algebraic branching programs (ABP), which we define now.

▶ **Definition 1** (Algebraic Branching Programs (ABP))**.** An algebraic branching program in variables $\{x_1, x_2, \ldots, x_n\}$ over a field $\mathbb{F}$ is a directed acyclic graph with a designated *starting vertex* s with in degree zero, a designated *end vertex* t with out degree zero, and the edge between any two vertices is labeled by an affine form from $\mathbb{F}[x_1, x_2, \ldots, x_n]$.

We say that the ABP is homogeneous, if the polynomial computed at every vertex is a homogeneous polynomial.

The weight of any (directed) path in an ABP is the product of labels of the edges in the path. The polynomial computed at a particular vertex v is the sum of weights of all paths from the starting vertex s to v. The polynomial computed by the ABP is the polynomial computed at the end vertex t.

In terms of their power of computation, ABPs lie somewhere between arithmetic formula and general arithmetic circuits, in the following precise sense. An arithmetic formula can be converted into an ABP such that the number of vertices in the ABP is at most the number of vertices in the formula. On the other hand, an ABP can be transformed into an arithmetic circuit such that the number of vertices in the circuit is at most the sum of the number of edges and the number of vertices in the ABP[1]. Since arithmetic formula and ABP seem to be weaker models of computation than general arithmetic circuits, it is conceivable that proving lower bounds for them could be a more tractable challenge than proving lower bounds for general arithmetic circuits. In a way, this reflects in the current state of art where we know almost quadratic lower bounds for arithmetic formula [7, 13], whereas the best lower bounds known for arithmetic circuits or even ABPs continue to be the weakly super-linear [16, 2]. Moreover, to the best of our knowledge, even for *homogeneous* ABPs, prior to the results in this paper, no non-trivial super-linear lower bounds seem to be known on the number of vertices, whereas for the number of edges, the results in [16, 2] give an $\Omega(n \log n)$ lower bound[2]. We remark that in the setting of boolean circuit complexity it is possible to extend the formula lower bound of Nechiporuk [9] to show an $\Omega(n^{1+\epsilon})$ lower bounds for both deterministic and non-deterministic branching programs. However, such an extension is not known in the algebraic setting. The key difference stems from the fact that the edge labels for a boolean branching program are just individual literals or constants, as opposed to arbitrary affine forms as in the case of an algebraic branching program. And, indeed if we restrict Definition 1 so that every edge label is a field constant or an affine form in a single variable (and not a general affine form), then the formula lower bounds of Kalorkoti [7] do extend to such special cases and give a super-linear lower bound on the number of edges in an ABP. However, transforming a general ABP given by Definition 1 to this form seems to incur a blowup of factor n in the number of edges, and it is unclear if something non-trivial can be recovered via this approach.

We would like to remark that even though not much seems to be known for lower bounds for general algebraic branching programs, much progress has been made on the understanding

[1] These transformations also preserve homogeneity.

[2] Note that if an ABP computes a degree d polynomial, it must have at least $d+1$ vertices, since every edge contributes degree at most 1, and there must be a path with at least d edges to push the degree up to d. So, we think of this lower bound of $\Omega(d)$ on an ABP as *trivial*. Because of this, whenever we mention a (homogeneous) ABP lower bound for an n variate polynomial of degree d, we think of $d \leq n$, so that the trivial lower bound of d is at most linear in n.

of many restricted and more structured variants of algebraic branching programs; both from the point of view of lower bounds and deterministic polynomial identity testing. For instance, strongly superpolynomial lower bounds are known for non-commutative ABPs [10] and read k-oblivious ABPs [1]. For an overview of known polynomial identity testing results for read once oblivious algebraic branching programs, we refer the reader to the PhD thesis of Michael Forbes [6].

In this paper, we study the question of proving an improved lower bound for general algebraic branching programs. Our main result is a quadratic lower bound on the number of vertices for a general *homogeneous* ABP. To the best of our knowledge, this is the first such non-trivial superlinear lower bound. Also, this immediately implies a quadratic lower bound on the number of edges, improving the earlier bound of $\Omega(n \log n)$ [16, 2]. We now precisely state the theorem.

▶ **Theorem 2.** *Let $\mathbb{F}$ be a field of characteristic zero or relatively prime to d. Let B be a homogeneous algebraic branching program over the field $\mathbb{F}$ which computes the polynomial $P_{(n,d)}(\mathbf{x})$. Then, the number of vertices in B is at least $\Omega(nd)$.*

▶ Remark. Theorem 2 holds for a slightly more general class of branching programs than homogeneous branching programs. Our proof continues to hold if the number of non-trivial affine linear forms on any path from the start vertex s to the end vertex t is at most the degree of the polynomial computed. For our proofs, we consider this slightly more general model. In some sense, this generalization is a more natural model to study since the model is closed under affine transformations.

Picking $d = \Theta(n)$ would give us the desired quadratic lower bound. Based on the known results, there are two natural approaches to try for ABP lower bounds. The first would be to try and extend the proof of formula lower bounds in [7] to a general ABP. It is not clear if this approach can be made to work[3]. One major obstacle seems to be that the edge labels in the ABP are general affine forms, which seems to make it tricky to analyse the complexity measure used in [7] for an ABP. Another approach would be try and use the special structure of an ABP, and aim to get an improved analysis of the circuit lower bound obtained in [16, 2]. It is unclear to us if the original proofs in [16, 2] can be used to this end. One of the challenges with adapting the proofs in [16, 2] to obtain better ABP lower bounds seems to be that in the obvious conversion of an ABP to a circuit, the number of vertices in the circuit obtained is the sum of the number of vertices and the number of edges in the ABP. It seems tricky to extract any non-trivial bound on the number of vertices of the ABP from this transformation since the degree of every vertex in an ABP is unbounded in general. Even in the setting of number of edges, it is not apriori clear if a better lower bound can be proved using the proof in [16, 2].

For our proof in this paper, we essentially follow this high level strategy. On the way, we give an alternate proof of an $\Omega(n \log n)$ lower bound for homogeneous arithmetic circuits. The ideas in this proof turn out to be a bit more malleable and sensitive to the underlying model of computation than the original one, and indeed for a homogeneous ABP we obtain a better lower bound by a direct analysis which crucially relies on the structure of the ABP. Formally, we give an alternate proof of the following result.

[3] However, we do not know how to formally show that there is a nearly linear size ABP which has high complexity in terms of the measure used in [7].

▶ **Theorem 3** ([16, 2]). *Let $\mathbb{F}$ be a field of characteristic zero or relatively prime to d. Then, any homogeneous arithmetic circuit which computes the polynomial $P_{(n,d)}$ has at least $\Omega(n \log d)$ gates.*

The statement above is a special case of a classical result [16, 2], where they show a similar lower bound for all (not necessarily homogeneous) arithmetic circuits. For the original proof, Baur & Strassen [2] showed that if an n variate polynomial can be computed by an arithmetic circuit of size s, then all its partial derivatives can be computed by a multi-output circuit of size $O(s)$. They combined this structural result with an $\Omega(n \log d)$ lower bound on the size of multi-output arithmetic circuits, proved by Strassen [16]. Strassen's proof, in turn relies on a beautiful application of Bezout's theorem. Our proof does not rely on the Bezout's theorem directly but uses some other elementary properties of algebraic varieties. We enumerate the properties used in Section 2. It is not clear to us if our proof is any more elementary than the proof in [16, 2] or vice versa, although as we alluded to, it does seem to be more flexible to the underlying model than the original proof.

In a short and beautiful paper, Smolensky [15] gave a completely elementary proof of the $\Omega(n \log n)$ lower bound for general circuits. Smolensky's proof uses just elementary linear algebra, and therefore is definitely simpler than our proof of Theorem 3. However, it is not clear if this proof can be strengthened to show Theorem 2.

Determinantal Complexity

Another well known model of computation in algebraic complexity theory, which is relevant to the results in this paper is the notion of *determinantal complexity*, defined as follows.

▶ **Definition 4** (Determinantal complexity). Let $P \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ be a polynomial of degree d. The determinantal complexity of P is the smallest k such that there is a $k \times k$ matrix M with entries being affine forms in $\mathbb{F}[x_1, x_2, \ldots, x_n]$ such that $\text{Determinant}(M) = P$.

Perhaps not surprisingly, the state of known lower bounds on determinantal complexity is also fairly modest, with the best lower bound known being an $\frac{n}{2}$ lower bound for an n variate polynomial family [8, 4]. Over the field of real numbers, this bound was recently improved to n by Yabe [17].

We now state our last result where we give a simple proof of the lower bound for determinantal complexity of $P_{(n,d)}$.

▶ **Theorem 5** ([8, 4]). *Let $\mathbb{F}$ be a field of characteristic zero or characteristic $p \neq 2$ such that $2 \leq d < p$. Then, the determinantal complexity of $P_{(n,d)}$ over the field $\mathbb{F}$ is at least $n/2$.*

The original proofs of Theorem 5 of an $n/2$ lower bound on the determinantal complexity of the permanent of an $\sqrt{n} \times \sqrt{n}$ matrix due to Mignon and Ressayre [8] over fields of characteristic zero, and due to Cai, Chen and Li [4] over all fields of characteristic not equal to 2, both rely on analysing the rank of the Hessian matrix associated to the permanent. On the other hand, for our proof, we will formulate a criterion for proving determinantal complexity lower bound upto $n - o(n)$ for an n variate polynomial using elementary linear algebra. This part of the proof is completely elementary. We then show that the polynomial $P_{(n,d)}$ satisfies this criterion for some weaker choice of parameters. Also, our argument essentially remains the same over all fields. Interestingly, over the field of real numbers, we get a lower bound of n for $P_{(n,d)}$ as long as d is even. This matches an improvement of factor 2 shown recently by Yabe [17] for the determinantal complexity of the permanent over the field of real numbers. For the reals, our proof of an n lower bound turns out to be extremely simple.

Proof outline

The proofs of all the three theorems crucially rely on a structural property of the polynomial $P_{(n,d)}$, which we summarize in Lemma 6. A special case of this lemma, (see Corollary 14) is already quite interesting and sufficient for the *homogeneous* ABP and circuit lower bound proofs and appears to be known [12][4]. Our proof is along similar lines, but needs some more ideas.

▶ **Lemma 6.** *Let $\mathbb{F}$ be an algebraically closed field of characteristic zero or relatively prime to d. Let $\{Q_1, Q_2, \ldots, Q_k, R_1, R_2, \ldots, R_k\}$ be a set of polynomials in $\mathbb{F}[\mathbf{x}]$ such that the set of their common zeros $V = \mathbb{V}(Q_1, Q_2, \ldots, Q_k, R_1, R_2 \ldots, R_k)$ is non-empty. Let P be any polynomial in $\mathbb{F}[\mathbf{x}]$ of degree at most $d-1$, such that*

$$P_{(n,d)} = P + \sum_{i=1}^{k} Q_i \cdot R_i \,.$$

Then, $k \geq n/2$.

For the proofs of the main theorems, we use the linear algebraic and combinatorial structure of the models at hand (namely homogeneous ABP, homogeneous circuits and determinantal complexity) to reduce to an application of Lemma 6. Proofs of Theorem 2, Theorem 3 rely on multiple applications of Lemma 6, while the proof of Theorem 5 relies on a single application of a very special case of Lemma 6, which in itself has a very simple proof. The proof of Lemma 6 requires some properties of the dimension of varieties defined by polynomials of a special form, and we give a simple (though not completely self contained[5]) proof in Section 3.1.

Theorem 3 and Theorem 5 are two fundamental lower bounds in algebraic complexity theory and have been at the frontier of our understanding of lower bounds for these models for the past many years. Improving these bounds is perhaps one of the most important open problems in this line of research. Therefore, it seems desirable to have newer and alternative proofs of these results. Moreover, the original proofs of Theorem 5 and Theorem 3 were quite different from each other and the results themselves were proved almost two decades apart. On the other hand, it is interesting to note that the proofs in this paper give essentially unified arguments for both these statements, as well as for homogeneous ABP lower bounds (even though we can show a super-linear lower bound only for homogeneous arithmetic circuits). We would also like to remark that since the proof of Lemma 6 relies on the dimension of varieties, a quantity always upper bounded by the number of variables, it appears likely that we would need new ideas to push the lower bound on k to anything larger than n.

Organization of the paper

We set up some notations and preliminaries in Section 2 and prove some technical claims needed for the proofs in Section 3.1. We prove Theorem 2 and Theorem 3 in Section 3.2 and Theorem 5 in Section 3.3.

[4] Saptharishi attributes the proof to Kayal.
[5] The proof uses some known standard properties of algebraic varieties, which we do not prove here.

2 Preliminaries

We now list some notations that we follow.

- $\mathbb{F}$ denotes a general field, and $\mathbb{C}$ denotes the field of complex numbers.
- Without loss of generality, for the results in this paper, we think of the field $\mathbb{F}$ to be algebraically closed. This is because an arithmetic circuit, a branching program or a matrix over a field $\mathbb{F}$ can be viewed to be over the algebraic closure of $\mathbb{F}$.
- The degree of a monomial $x_i^{e_1}x_2^{e_2}\cdots x_n^{e_n}$ is defined to be equal to $\sum_{i=1}^{n} e_i$.
- The degree of a polynomial P is the degree of the highest degree monomial in P with a non-zero coefficient.
- We denote the set $\{x_1, x_2, \ldots, x_n\}$ by the set $\mathbf{x}$.
- We denote the set $\{1, 2, 3, \ldots, t\}$ by $[t]$.
- An affine form in $\mathbb{F}[\mathbf{x}]$ is a polynomial of the form $\alpha_0+\sum_{i=1}^{n}\alpha_i x_i$, where $\alpha_0, \alpha_1, \alpha_2, \ldots, \alpha_n \in \mathbb{F}$.
- We say that a polynomial P has *no constant term* if the homogeneous component of degree 0 of P is 0. In particular, for any polynomial $P \in \mathbb{F}[\mathbf{x}]$ with no constant term, $P(0, 0, \ldots, 0) = 0$.
- For a square matrix M, we denote the determinant of M by $\det(M)$.
- For every gate (or vertex) g in an arithmetic circuit or an algebraic branching program, we denote by $[g]$, the polynomial computed at g. For the starting vertex s of an ABP, we define the polynomial computed at s, denoted by $[s]$ to be 1.
- For any set $\{Q_1, Q_2, \ldots, Q_t\}$ of polynomials in $\mathbb{F}[\mathbf{x}]$, we denote by $\mathbb{V}(Q_1, Q_2, \ldots, Q_t)$ the affine variety (or simply variety) of $Q_1, Q_2, \ldots, Q_t$ in $\mathbb{F}^n$, which is defined as follows:

$$\mathbb{V}(Q_1, Q_2, \ldots, Q_t) = \{\mathbf{a} \in \mathbb{F}^n : \forall i \in [t], Q_i(\mathbf{a}) = 0\} .$$

- For any set $\{Q_1, Q_2, \ldots, Q_t\}$ of polynomials in $\mathbb{F}[\mathbf{x}]$, we define the ideal generated by $Q_1, Q_2, \ldots, Q_t$ defined as follows:

$$\mathbb{I}(Q_1, Q_2, \ldots, Q_t) = \left\{ \sum_{i=1}^{t} R_i \cdot Q_i : \forall i \in [t], R_i \in \mathbb{F}[\mathbf{x}] \right\} .$$

- For any variety $V \subseteq \mathbb{F}^n$, we define the ideal associated to this variety, denoted by $\mathbb{I}(V)$ as follows:

$$\mathbb{I}(V) = \{R : R \in \mathbb{F}[\mathbf{x}], \text{ and } \forall \mathbf{a} \in V, R(\mathbf{a}) = 0\} .$$

Algebraic branching programs, arithmetic circuits and determinantal complexity

We have already defined an algebraic branching program and determinantal complexity in Section 1.

We now recall the definition of an arithmetic circuit.

▶ **Definition 7** (Arithmetic circuits)**.** An arithmetic circuit on variables $\mathbf{x}$ over a field $\mathbb{F}$ is a directed acyclic graph, where the vertices (also called gates) with in-degree zero (called input gates or leaves) are labeled either by constants over $\mathbb{F}$ or with variables in $\mathbf{x}$. The internal vertices all have in-degree (or fan-in) 2 and are labeled by $+$ or $\times$, which indicate summation and multiplication operations over the field $\mathbb{F}$. The edges feeding into a $+$ gate can be labeled by field constants.

An arithmetic circuit formally computes a polynomial in the natural way. A circuit is said to be *homogeneous* if the polynomial computed at every vertex in the circuit is a homogeneous polynomial. The number of vertices in a circuit is the size of the circuit. Since we restrict ourselves to fan-in two circuits in this paper, the number of edges and the number of vertices are within a constant factor of each other. We refer the reader to the excellent survey by Shpilka and Yehudayoff [13] for an introduction to arithmetic circuits, and an over view of prior work in this area.

For an algebraic branching program, we note that the number of vertices and the number of edges need not be within a constant fraction of each other, since the in-degree and out-degree of internal vertices is both unrestricted. In this sense, a super-linear lower bound on the number of edges in an ABP need not necessarily imply a super-linear lower bound on the number of vertices. We also remark that that without loss of generality, we can assume that the underlying graph of an ABP is simple, i.e there is at most one edge between any pair of vertices. This follows from the fact that multiple edges can be combined into a single edge whose weight is the sum of weight of the original edges. Since edge weights are allowed to be arbitrary affine forms, this is a valid transformation for an ABP.

Ideals and varieties

A useful notion for our proofs will be that of an affine variety (or simply variety). For a field $\mathbb{F}$, a variety $V \subseteq \mathbb{F}^n$ is simply the set of common zeros of a set of polynomials in $\mathbb{F}[x_1, x_2, \ldots, x_n]$. Another relevant notion is the notion of an ideal. For a variety V, the ideal associated to V, denoted by $\mathbb{I}(V)$ is the set of all polynomials in $\mathbb{F}[\mathbf{x}]$ which vanish on V.

A fundamental property associated to an affine variety is its *dimension*, which takes a value between 0 and n. We do not formally define this, but this can be thought of as an appropriate generalization of the notion of dimension for linear spaces.

We refer the reader to the book by Cox, Little and O'Shea [5] for more on algebraic varieties and ideals and connections between them. For the proofs in this paper, we will rely on the following properties of dimension of a variety.

▶ **Lemma 8** (Section 2.8 in [14]). *Let S be a set of polynomials in n variables over an algebraically closed field $\mathbb{F}$ such that $|S| \leq n$. Let $V = \mathbb{V}(S)$ be the set of common zeros of polynomials in S.*

$$V = \{\mathbf{a} \in \mathbb{F}^n : \forall f \in S, f(\mathbf{a}) = 0\} .$$

If V is non-empty, then, the dimension of $V(S)$ is at least $n - |S|$.

The following two facts are basic properties of the dimension of a variety and can be found in Section 4 of Chapter 9 in [5].

▶ **Lemma 9.** *Let $\mathbb{F}$ be an algebraically closed field, and let $V_1 \subseteq \mathbb{F}^n$ and $V_2 \subseteq \mathbb{F}^n$ be two affine varieties such that $V_1 \subseteq V_2$. Then, the dimension of V_1 is at most the dimension of V_2.*

▶ **Lemma 10.** *Let $\mathbb{F}$ be an algebraically closed field and let $V \subseteq \mathbb{F}^n$ be an affine variety. Then, the dimension of V is zero if and only if V is finite.*

3 Proofs of main theorems

We now proceed to prove the results. We start with a technical lemma, which proves to be critical for all our main results. A special case of the lemma where each of the polynomials

Q_i and R_i is homogeneous and the polynomial P is identically zero seems to be known [12]. The statement in Lemma 6 is a generalization of this special case. The proof is along very similar lines, but needs a few more ideas.

3.1 Technical claims

For this section, we work over the field $\mathbb{C}$ of complex numbers, but the results continue to hold over any algebraically closed field of characteristic p such that p does not divide the parameter d. This ensures that certain partial derivatives which come up in the proofs do not vanish. We start with the following lemma.

▶ **Lemma 11** (Restatement of Lemma 6). *Let $\{Q_1, Q_2, \ldots, Q_k, R_1, R_2, \ldots, R_k\}$ be a set of polynomials in $\mathbb{C}[\mathbf{x}]$ such that the set of their common zeros $V = \mathbb{V}(Q_1, Q_2, \ldots, Q_k, R_1, R_2 \ldots, R_k)$ is non-empty. Let P be any polynomial in $\mathbb{C}[\mathbf{x}]$ of degree at most $d-1$, such that*

$$P_{(n,d)} = P + \sum_{i=1}^{k} Q_i \cdot R_i \,.$$

Then, $k \geq n/2$.

Proof. We prove the lemma via contradiction. If possible, let $k < n/2$. This implies that $n - 2k > 0$. From the hypothesis of the lemma, $\mathbb{V}(Q_1, Q_2, \ldots, Q_k, R_1, R_2, \ldots, R_k)$ is non-empty. Therefore, by Lemma 8, the dimension of $\mathbb{V}(Q_1, Q_2, \ldots, Q_k, R_1, R_2, \ldots, R_k)$ is at least $n - 2k > 0$.

For any variable $x_j \in \mathbf{x}$, observe that

$$\frac{\partial P_{(n,d)}}{\partial x_j} = \frac{\partial P}{\partial x_j} + \sum_{i=1}^{k} \frac{\partial Q_i}{\partial x_j} \cdot R_i + \sum_{i=1}^{k} Q_i \cdot \frac{\partial R_i}{\partial x_j} \,.$$

This implies that

$$dx_j^{d-1} - \frac{\partial P}{\partial x_j} = \sum_{i=1}^{k} \frac{\partial Q_i}{\partial x_j} \cdot R_i + \sum_{i=1}^{k} Q_i \cdot \frac{\partial R_i}{\partial x_j} \,.$$

It is easy to see that for every $x_j \in \mathbf{x}$, the right hand side in the equality above vanishes on every point in $\mathbb{V}(Q_1, Q_2, \ldots, Q_k, R_1, R_2, \ldots, R_k)$. Therefore,

$$\mathbb{V}(Q_1, Q_2, \ldots, Q_k, R_1, R_2, \ldots, R_k) \subseteq \mathbb{V}\left(\left\{dx_j^{d-1} - \frac{\partial P}{\partial x_j} : x_j \in \mathbf{x}\right\}\right) .$$

In particular, Lemma 9 implies that the dimension of $\mathbb{V}\left(\left\{dx_j^{d-1} - \frac{\partial P}{\partial x_j} : x_j \in \mathbf{x}\right\}\right)$ is at least $n - 2k$. Since P is a polynomial of degree at most $d-1$, each first order partial derivative of P is of degree at most $d-2$. Now, it follows from Lemma 12, (which we prove below) that the dimension of $\mathbb{V}\left(\left\{dx_j^{d-1} - \frac{\partial P}{\partial x_j} : x_j \in \mathbf{x}\right\}\right)$ is zero. Therefore, $n - 2k \leq 0$, but this is a contradiction for $k < n/2$. ◀

▶ **Lemma 12.** *Let d be a positive natural number. For every choice of polynomials $g_1, g_2, \ldots, g_n \in \mathbb{C}[\mathbf{x}]$ of degree at most $d-1$, the dimension of the variety $\mathbb{V}(x_1^d - g_1, x_2^d - g_2, \ldots, x_n^d - g_n)$ is zero.*

Proof. Let $V = \mathbb{V}(x_1^d - g_1, x_2^d - g_2, \ldots, x_n^d - g_n)$. To prove the lemma, we use Lemma 10. We show that the cardinality of V is at most $T = \binom{n+n(d-1)}{n}$. We prove this via contradiction. If the cardinality of V is larger than T, then we focus our attention on an arbitrary subset $S \subseteq V$ of size equal to $T+1$. Now, consider the linear space of polynomial functions from S to $\mathbb{C}$. Clearly, the dimension of this linear space must be at least $T+1$, since the indicator function of every point in S can be expressed as a sufficiently high degree polynomial and these polynomials are linearly independent. We now argue that the dimension of the linear space of all polynomial functions from V to $\mathbb{C}$ (and therefore from S to C) is upper bounded by T. This completes the proof by contradiction. To this end, we prove the following claim. Let $I = \mathbb{I}(V)$ be the ideal corresponding to V. Clearly, for every $i \in [n]$, $x_i^d - g_i \in I$.

▶ **Claim 13.** *Let P be any polynomial in $\mathbb{C}[\mathbf{x}]$ of degree Δ strictly larger than $n(d-1)$. Then, there exists a polynomial P'' of degree at most $n(d-1)$ and polynomials $h_1, h_2, \ldots, h_n$ such that*

$$P = P'' + \sum_{i=1}^{n} (x_i^d - g_i) \cdot h_i$$

Proof. Note that for every $x_i \in \mathbf{x}$, the polynomial P' obtained from P by replacing every occurrence of x_i^d by g_i is equivalent to P mod the ideal I, since $P - P'$ is divisible by $x_i^d - g_i$, which is in the ideal. So, we can keep performing this replacement while still maintaining equivalence modulo the ideal I. Note that the process terminates eventually, since x_i^d is being replaced by a polynomial of strictly smaller degree. Let P'' be the polynomial obtained when the process terminates. It follows that the individual degree of every variable x_i in P'' is upper bounded by $d-1$, and hence the degree of P'' is at most $n(d-1)$. This proves the claim. ◀

Therefore, the space of all polynomial functions from V to $\mathbb{C}$ is spanned by a subset of polynomials in $\mathbb{C}[\mathbf{x}]$ of degree at most $n(d-1)$. Hence, the dimension of this linear space is at most the number of monomials of degree at most $n(d-1)$ in n variables, which is equal to T. ◀

The following corollary of Lemma 6 is already interesting and seems to be well known [12].

▶ **Corollary 14.** *For every set $\{Q_1, Q_2, \ldots, Q_k, R_1, R_2, \ldots, R_k\}$ of homogeneous polynomials of degree at least 1, if*

$$P_{(n,d)} = \sum_{i=1}^{k} Q_i \cdot R_i \,.$$

Then, $k \geq n/2$.

3.2 Lower bound for homogeneous algebraic branching programs

In this section, we prove Theorem 2. We will in fact show that the theorem is true for a class of algebraic branching programs which are slightly more general than homogeneous ABPs. We say that an ABP has *formal degree* at most d, if the number of non-constant edge weights on any path from s to t is at most d. In general, we define the formal degree of any vertex v in an ABP to be the maximum number of non-constant edge weights along any path from s to v. We first argue that we can convert a homogeneous ABP computing a polynomial of degree d to an ABP of formal degree d.

▶ **Lemma 15.** *Let B be a homogeneous ABP with r vertices which computes a homogeneous polynomial P of degree d. Then, there is an ABP B' computing P such that B' has at most r vertices and has formal degree at most d.*

We defer the proof of this lemma to the end of this section, and use it to complete the proof of Theorem 2. We now prove the following structural lemma for ABPs of formal degree d.

▶ **Lemma 16.** *Let B be an algebraic branching program of formal degree at most d with b vertices, which computes an n-variate polynomial P of degree d. For any $i \in \{1, 2, 3, \ldots, d-1\}$, let $S_i = \{u_1, u_2, \ldots, u_m\}$ be the set of all vertices in B which compute a polynomial of degree equal to i. Then, there exist polynomials $h_1, h_2, \ldots, h_m$ and R of degree at most $d-1$ such that*

$$P = \sum_{j=1}^{m} [u_j] \cdot h_j + R$$

Proof. Let us consider all paths from the starting vertex s of B to the end vertex t of B which passes through some $u_j \in S_i$. The polynomial computed by the sum of weights of only these paths can be written as $[u_j] \cdot h_j$ where h_j is the polynomial given by the sum of weights of all paths from u_j to t. Now, we claim that the degree of h_j is at most $d - d_{u_j}$. This follows from the fact that if the degree of h_j was larger than $d - d_{u_j}$, then the formal degree of t will be larger than d which would contradict the hypothesis that B is of formal degree at most d.

We now use this observation to complete the proof of the lemma. Without loss of generality, let us assume that the vertices $u_1, u_2, \ldots, u_m$ are ordered in such a way that there is no directed path from u_j to $u_{j'}$ for any $j' > j$. We prove the following claim by a simple induction.

▶ **Claim 17.** *Fix any $j \in \{1, 2, 3, \ldots, m\}$. Then, there exists polynomials $h_1, h_2, \ldots, h_j$ of degree at most $d-1$ and a polynomial R_j computed by the ABP B'_j obtained from B by deleting all the vertices in $\{u_1, u_2, \ldots, u_j\}$ such that*

$$P = \sum_{k=1}^{j} [u_k] \cdot h_k + R_j \,.$$

Proof. For $k = 1$ the proof follows from the observation above. For the induction step, observe that in the ABP obtained by deleting the vertices $u_1, u_2, \ldots, u_k$, the polynomial computed by the vertex u_{k+1} is the same as the polynomial computed by the vertex u_{k+1} in the original ABP B. This is true since by our ordering of vertices $u_1, u_2, \ldots, u_m$ there are no directed paths from u_ℓ to $u_{\ell'}$ for any $\ell' > \ell$ in B. ◀

We now argue that the degree of R_m in Claim 17 is at most $d-1$. This would complete the proof of the lemma. Let B' be the ABP obtained from B by deleting all vertices in the set S_i in B. We know that R_t is the polynomial computed by B'. Let us consider any path $s, v_1, v_2, \ldots, v_k, t$ from s to t in B'. Note that all these vertices appear in the original ABP B. Let us consider the minimum j such that v_j has degree at least $i+1$ in B. Observe that the degree of v_{j-1} in B must be at most $i-1$, since we have deleted the vertices in S_i. Therefore, the degree of the monomials in the weight of the path $s, v_1, v_2, \ldots, v_k, t$ is at most $i - 1 + \ell + 1$ where ℓ is the maximum number of non-constant edge weights on any path from v_j to t in B. Now, observe that ℓ is at most $d - i - 1$. This is true since if $\ell \geq d - i$, then there would be a path in B from s to t through v_j such that there are at least $d+1$

non-constant edge weights on this path, thereby contradicting the hypothesis that the formal degree of B is at most d. ◀

We are now ready to complete the proof of Theorem 2.

▶ **Theorem 18** (Restatement of Theorem 2). *Let B be an algebraic branching program of formal degree at most d over $\mathbb{C}$ which computes the polynomial $P_{(n,d)}(\mathbf{x})$. Then, the number of vertices in B is at least $\Omega(nd)$.*

Proof. We partition the set of vertices in the ABP B, into $\Omega(d)$ many sets based on their degree. Then, we argue that each of these sets must have at least $n/2$ vertices. For $i \in \{1, \ldots, d-1\}$, let the set $S_i = \{u_1, u_2, \ldots, u_{t_i}\}$ be the set of all vertices in B which compute a polynomial of degree equal to i. From Lemma 16, we know that there are polynomials $h_{i,1}, h_{i,2}, \ldots, h_{i,t_i}$ and R_i of degree at most $d-1$ such that

$$P_{(n,d)} = \sum_{j=1}^{t_i} [u_j] \cdot h_{i,j} + R_i \,.$$

Let $[u_j]$ and $h_{i,j}$ be written as $[u_j] = [u_j]' + \alpha$ and $h_{i,j} = h'_{i,j} + \beta$ where α, β are constants and $[u_j]'$, $h'_{i,j}$ have no constant terms. Then,

$$[u_j] \cdot h_j = [u_j]' \cdot h'_{i,j} + Q_j$$

where Q_j has degree at most $d-1$. Therefore, without loss of generality, we get that there polynomials $h'_{i,1}, h'_{i,2}, \ldots, h'_{i,t_i}$ and R'_i such that

$$P_{(n,d)} = R'_i + \sum_{j=1}^{t_i} [u_j]' \cdot h'_{i,j}$$

where

- Degree of R'_i is most $d-1$.
- For every j, the constant term of each of the polynomials $[u_j]'$ and $h'_{i,j}$ is equal to zero and they have degree at least 1.

Note that since $[u_j]'$s and $h'_{i,j}$s have degree at least one and have no constant term, it follows that they vanish at the all zero point. In particular, $V = \mathbb{V}([u_1]', [u_2]', \ldots, [u_{t_i}]', h'_{i,1}, h'_{i,2}, \ldots, h'_{i,t_i})$ is non empty. So, by Lemma 6, it follows that t_i is at least $n/2$. Since this holds for all the $\Omega(d)$ values of i, and these sets S_i are all disjoint, this gives the desired lower bound on the number of vertices of B. ◀

We now prove Lemma 15.

Proof of Lemma 15. We will start with the ABP B and obtain an ABP B' by modifying or deleting some of the edge weights in B such that the polynomial computed by B' is the same as the polynomial computed by B. Moreover, B' will have the additional property that the degree of the homogeneous polynomial computed at every vertex v equals the formal degree of v. The proof will be via an induction, where we process vertices in the topological order, i.e we process a vertex v only after processing every vertex u such that (u, v) is an edge in B.

The base case of this induction is trivial as there is nothing to do for the starting vertex s.

For the induction step, we process a vertex v. Let $u_1, u_2, \ldots, u_m$ be all the vertices such that (u_j, v) is an edge in B. Let the weight of (u_j, v) be $\ell_j + \alpha_j$, where ℓ_j is a homogeneous

linear form (which could be identically zero) and α_j is a constant. Also, let d_{u_j} be the degree of $[u_j]$. So, we have the following identity:

$$[v] = \sum_{j=1}^{m} [u_j] \cdot (\ell_j + \alpha_j) \,.$$

We separate out the u_js based on their degree.

$$[v] = \sum_{j:d_v < d_{u_j}} [u_j] \cdot (\ell_j + \alpha_j) + \sum_{j:d_v = d_{u_j}} [u_j] \cdot (\ell_j + \alpha_j) + \sum_{j:d_v > d_{u_j}} [u_j] \cdot (\ell_j + \alpha_j) \,.$$

We now observe that since the polynomial computed at v and every u_j is homogeneous, and $[v]$ has degree d_v, the following identity is also true.

$$[v] = \sum_{j:d_v < d_{u_j}} [u_j] \cdot 0 + \sum_{j:d_v = d_{u_j}} [u_j] \cdot (\alpha_j) + \sum_{j:d_v > d_{u_j}} [u_j] \cdot (\ell_j + \alpha_j) \,.$$

So, in B', we replace the edge weights as follows:

- For every vertex u_j such that $d_{u_j} > d_v$, we delete the edge (u_j, v), and
- for every vertex u_j such that $d_{u_j} = d_v$ with the edge (u_j, v) having weight $\ell_j + \alpha_j$, we relabel it with α_j. ◀

3.2.1 Lower bound for homogeneous arithmetic circuits

The proof of Theorem 3 is along the lines of the proof of Theorem 2 that we described above. The main difference is that we partition the set of vertices in the circuit into $\Omega(\log d)$ sets based on their degrees defined as follows. For $i \in \{1, 2, \ldots, \log(d) - 1\}$, we define the set S_i to be the set of all vertices v in a homogeneous circuit C such that the degree d_v of the polynomial computed at v satisfies $2^i \leq d_v < 2^{i+1} - 1$. For this definition of the set S_i, a structural lemma analogous to Lemma 16 is true, and is easy to prove. Combining this with Lemma 6, would imply that the size of S_i is at least $\Omega(n)$. Since there are $\log d$ such sets, we get a bound of $\Omega(n \log d)$. We skip the rest of the details.

3.3 Lower bound on determinantal complexity

In this section, we complete the proof of Theorem 5. We start by proving the following lemma.

▶ **Lemma 19.** *Let $Q \in \mathbb{F}[\mathbf{x}]$ be a homogeneous polynomial of degree d. Let M be a $t \times t$ matrix, whose entries are affine forms in the variables $\mathbf{x}$, such that*

$$\det[M] = Q \,.$$

Then, there exists a linear subspace S of dimension at least $n - t$, such that $Q(\mathbf{a}) = 0, \forall \mathbf{a} \in S$.

Proof. Since the entries of M are affine functions in the variables in $\mathbf{x}$, we can write M as

$$M(\mathbf{x}) = M_0 + \sum_{i=1}^{n} M_i x_i \,.$$

Here, $M_0, M_1, \ldots, M_n$ are $t \times t$ matrices over $\mathbb{F}$. Since Q is homogeneous, it follows that

$$Q(0, 0, \ldots, 0) = 0,$$

it follows that $\det[M_0] = 0$. Therefore, M_0 is not full rank. Hence, there is a non-zero vector $v \in \mathbb{F}^t$, which is in the kernel of v, i.e $M_0 v = 0$. Let us consider the set $S \subseteq \mathbb{F}^n$, defined as

$$S = \left\{ (a_1, a_2, \ldots, a_n) \in \mathbb{F}^n : \left(\sum_{i=1}^{n} M_i a_i \right) \cdot v = 0 \right\} .$$

In other words, S is the set of all vectors $\mathbf{a}$ in $\mathbb{F}^n$ such that the vector v is in the kernel of $\sum_{i=1}^{n} M_i a_i$. Observe that this implies that v is in the kernel of $M(\mathbf{a})$, since it is already in the kernel of M_0, by choice. Thus, $M(\mathbf{a})$ is rank deficient for $\mathbf{a} \in S$. Hence, $\det(M(\mathbf{a})) = 0$ for every $\mathbf{a} \in S$. Moreover, since S is a linear space of dimension at least $n - t$, it follows that Q is zero on every point on a subspace of dimension at least $n - t$. ◀

Observe that from the degree requirements, it follows that the determinantal complexity of a degree d polynomial is at least d. Hence, if we can construct an explicit polynomial of degree $d = o(n)$ such that it does not vanish on any linear subspace of dimension larger than $k(n, d)$, then from Lemma 19, we will obtain a lower bound of $n - k$. It is known that at least over small fields a random homogeneous polynomial of degree d in n variables does not vanish on any affine subspace of dimension much larger than $n^{O(1/d)}$ [3]. Therefore, in principle, d can be taken as small as $O(\log n)$ and $k = O(1)$ over such fields. The challenge is to construct such polynomial families explictly. Over small fields constructions of this nature are known, although the parameters seem to be far from what would be true for a random polynomial, see for example [3]. Even beyond the application to minor improvements in known determinantal complexity lower bounds, explicit construction of such *subspace evasive* polynomials is an extremely interesting open question.

We now observe that the polynomial $P_{(n,d)}$ already lets us recover the $n/2$ lower bound on determinantal complexity over the field of complex numbers and any field of characteristic p not equal to 2. For fields of characteristic equal to p, our proof would work, for instance if we pick d such that $2 \leq d < p$. In fact, over reals, we get a lower bound of n for $P_{(n,d)}$ for every even d. As alluded to in the introduction, such a lower bound of n was proved over reals by Yabe [17] for the permanent of an $\sqrt{n} \times \sqrt{n}$ matrix via a very different proof.

A useful notion for the rest of proof will be the notion of a *formal* restriction of a polynomial to a linear space, which is defined using the following observation.

▶ **Observation 20.** *Let $S \subseteq \mathbb{F}^n$ be any linear space of co-dimension equal to t and let P be any polynomial in $\mathbb{F}[\mathbf{x}]$. Then, there exists a subset V of variables $\mathbf{x}$ of size equal to $n - t$ and a polynomial Q_t depending only on the variables in V such that*

- *The degree of Q_t is at most the degree of P.*
- *For every $\mathbf{a} \in S$, $P(\mathbf{a}) = Q_t(\mathbf{a})$.*

Proof. Since S is a linear space of co-dimension t, it follows that there are coordinates $\{i_1, i_2, \ldots, i_t\}$ and linear forms $L_1, L_2, \ldots, L_t$ depending only on variables outside $\{i_1, i_2, \ldots, i_t\}$, such that

$$S = \left\{ \mathbf{a} \in \mathbb{F}^n : \forall j \in [t], a_{i_j} - L_j(\mathbf{a}) = 0 \right\} .$$

We define $V = \mathbf{x} \setminus \{x_{i_1}, x_{i_2}, \ldots, x_{i_t}\}$. Without loss of generality, we assume that $i_j = j$. Let Q_i be obtained from P by replacing the variables $x_1, x_2, \ldots, x_i$ in P by $L_1, L_2, \ldots, L_i$. By induction on i, it can be observed that

$$P - Q_i = \sum_{j=1}^{i} (x_j - L_j) \cdot R_j$$

where R_j is a polynomial of degree at most $d-1$. Moreover, by construction, Q_i does not depend on the variables $x_1, x_2, \ldots, x_i$. Now, from the definitions, we get that for any $\mathbf{a} \in S$,

$$P(\mathbf{a}) = Q(\mathbf{a})\,.$$

Since each Q_i is obtained from P by a linear transformation of the set of variables, the degree does not increase in the process. ◀

We call the polynomial Q_t obtained in the proof to be a formal restriction of P on S. We also get the following useful corollary.

▶ **Corollary 21.** *Let $\mathbb{F}$ be any field with at least $d+1$ elements, and let $P \in \mathbb{F}[\mathbf{x}]$ be any homogeneous polynomial of degree d. If S is a subspace of $\mathbb{F}^n$ of co-dimension t such that P evaluates to zero on S, then, there exist homogeneous linear forms $\ell_1, \ell_2, \ldots, \ell_t$ and homogeneous polynomials $R_1, R_2, \ldots, R_t$ of degree $d-1$ such that*

$$P = \sum_{i=1}^{t} \ell_i \cdot R_i\,.$$

Proof. The polynomial Q_t obtained in Observation 20 satisfies

$$P - Q_t = \sum_{i=1}^{t} (x_i - L_i) \cdot R_i$$

where each L_i is a homogeneous linear form, and each R_i is a homogeneous polynomial of degree $d-1$. Moreover, Q_t depends only on the un-restricted variables $x_{t+1}, x_{t+2}, \ldots, x_n$, and is of degree at most d, and for every $j \in \{1, 2, \ldots, t\}$ and $\mathbf{a} \in S$, $a_j = L_j(\mathbf{a})$. Since P evaluates to zero everywhere on S, it follows that $Q_t \in \mathbb{F}[x_{t+1}, x_{t+2}, \ldots, x_n]$ evaluates to zero everywhere on the grid $\mathbb{F} \times \mathbb{F} \times \ldots \times \mathbb{F}$. Since $\mathbb{F}$ has at least $d+1$ elements and Q_t is of degree at most d, by the Schwartz-Zippel lemma, Q_t must be identically zero. So,

$$P = \sum_{i=1}^{t} (x_i - L_i) \cdot R_i\,.$$ ◀

We now complete the proof of Theorem 5. We present the proof over the field of complex numbers, but it will be clear from the proof that the statement is true for any finite field of characteristic $p \neq 2$ such that the degree d of $P_{(n,d)}$ satisfies $2 \leq d < p$.

Proof of Theorem 5. Let M be a $t \times t$ matrix of affine forms over $\mathbb{C}[\mathbf{x}]$ such that

$$P_{(n,d)} = \det[M]$$

From Lemma 19, it follows that there is a linear subspace $S \in \mathbb{C}^n$ of dimension at least $n - t$ such that

$$P_{(n,d)}(\mathbf{a}) = 0, \forall \mathbf{a} \in S$$

From Corollary 21, it follows that there exist t homogeneous linear forms $\ell_1, \ell_2, \ldots, \ell_t$ and homogeneous polynomials $R_1, R_2, \ldots, R_t$ of degree equal to $d-1$, such that

$$P_{(n,d)} = \sum_{i=1}^{t} \ell_i \cdot R_i$$

From Lemma 6, we get that $t \geq n/2$. ◀

▶ Remark. Over reals, this argument gives a simple proof of the currently best lower bound of n for the polynomial $x_1^2 + x_2^2 + \ldots + x_n^2$ since this polynomial has exactly one zero in $\mathbb{R}^n$ and in particular, is not zero on any linear subspace of non-trivial dimension.

4 Open problems

We end with some open problems.

- The most interesting question here would be to extend the results here and prove a quadratic lower bound for general (possibly non-homogeneous) algebraic branching programs. Lemma 16 is not true for a general ABP and hence the proofs in this paper do not extend to the non-homogeneous setting.
- Another question of interest is to construct explicit polynomials of low degree which do not vanish on very large linear subspaces over all fields. Beyond the application to minor improvements in the determinantal complexity lower bounds, this seems to be a natural algebraic question.
- Of course, improving the lower bounds here is an extremely interesting problem. In fact, it is known that proving a super-quadratic lower bound for general algebraic branching programs implies a super-linear lower bound for determinantal complexity (see for example [17]). Perhaps the first step towards this goal could be to prove super-quadratic lower bound for homogeneous formulas. Currently, no such bounds are known.

Acknowledgments. I am thankful to Prahladh Harsha, Swastik Kopparty and Ramprasad Saptharishi for helpful discussions, and to Josh Grochow for pointing out a reference ([14]) for Lemma 8. Also, many thanks to Pooya Hatami and Mike Saks for sitting through a presentation of the proof and to Prahladh for comments on the writing which helped improve the presentation of the paper.

References

1 Matthew Anderson, Michael A. Forbes, Ramprasad Saptharishi, Amir Shpilka, and Ben Lee Volk. Identity testing and lower bounds for read-k oblivious algebraic branching programs. In *Proceedings of the 31st Annual Computational Complexity Conference (CCC 2016)*, volume 50 of *LIPIcs*, pages 30:1–30:25, 2016. doi:10.4230/LIPIcs.CCC.2016.30.

2 Walter Baur and Volker Strassen. The complexity of partial derivatives. *Theoretical Computer Science*, 22:317–330, 1983. doi:10.1016/0304-3975(83)90110-X.

3 Eli Ben-Sasson and Swastik Kopparty. Affine dispersers from subspace polynomials. *SIAM J. Comput.*, 41(4):880–914, 2012. doi:10.1137/110826254.

4 Jin-yi Cai, Xi Chen, and Dong Li. Quadratic lower bound for permanent vs. determinant in any characteristic. *Computational Complexity*, 19(1):37–56, 2010. doi:10.1007/s00037-009-0284-2.

5 David A. Cox, John B. Little, and Donal O'Shea. *Ideals, Varieties and Algorithms*. Undergraduate texts in mathematics. Springer, 2007. doi:10.1007/978-0-387-35651-8.

6 Michael A. Forbes. *Polynomial Identity Testing of Read-Once Oblivious Algebraic Branching Programs*. PhD thesis, Massachusetts Institute of Technology, 2014.

7 Kyriakos Kalorkoti. A Lower Bound for the Formula Size of Rational Functions. *SIAM Journal of Computing*, 14(3):678–687, 1985. doi:10.1137/0214050.

8 Thierry Mignon and Nicolas Ressayre. A quadratic bound for the determinant and permanent problem. *International Mathematics Research Notes*, 2004(79):4241–4253, 2004. doi:10.1155/S1073792804142566.

9 E. I. Nechiporuk. On a boolean function. *Soviet Math. Dokl.*, pages 999–1000, 1966.

10 Noam Nisan. Lower bounds for non-commutative computation. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing (STOC 1991)*, pages 410–418, 1991. doi:10.1.1.17.5067.

11 Ramprasad Saptharishi. A survey of lower bounds in arithmetic circuit complexity. Github survey, 2015. URL: https://github.com/dasarpmar/lowerbounds-survey/releases/.

12 Ramprasad Saptharishi. personal communication, 2016.

13 Amir Shpilka and Amir Yehudayoff. Arithmetic circuits: A survey of recent results and open questions. *Foundations and Trends in Theoretical Computer Science*, 5:207–388, March 2010. doi:10.1561/0400000039.

14 J. Smith. *Introduction to Algebraic Geometry*. Textbooks in Mathematics. Taylor & Francis, 2014.

15 Roman Smolensky. Easy lower bound for a strange computational model. *Computational Complexity*, 6(3):213–216, 1997. doi:10.1007/BF01294255.

16 V. Strassen. Die Berechnungskomplexiät von elementarsymmetrischen Funktionen und von Interpolationskoeffzienten. *Numerische Mathematik*, 20:238–251, 1973.

17 Akihiro Yabe. Bi-polynomial rank and determinantal complexity. *CoRR*, abs/1504.00151, 2015. URL: http://arxiv.org/abs/1504.00151.

On Algebraic Branching Programs of Small Width*

Karl Bringmann[1], Christian Ikenmeyer[2], and Jeroen Zuiddam[3]

1 **Max-Planck-Institut für Informatik, Saarland Informatics Campus, Saarbrücken, Germany**
kbringma@mpi-inf.mpg.de
2 **Max-Planck-Institut für Informatik, Saarland Informatics Campus, Saarbrücken, Germany**
cikenmey@mpi-inf.mpg.de
3 **Centrum Wiskunde & Informatica, Amsterdam, The Netherlands**
j.zuiddam@cwi.nl

Abstract

In 1979 Valiant showed that the complexity class $\mathbf{VP}_{\mathrm{e}}$ of families with polynomially bounded formula size is contained in the class $\mathbf{VP}_{\mathrm{s}}$ of families that have algebraic branching programs (ABPs) of polynomially bounded size. Motivated by the problem of separating these classes we study the topological closure $\overline{\mathbf{VP}_{\mathrm{e}}}$, i.e. the class of polynomials that can be approximated arbitrarily closely by polynomials in $\mathbf{VP}_{\mathrm{e}}$. We describe $\overline{\mathbf{VP}_{\mathrm{e}}}$ with a strikingly simple complete polynomial (in characteristic different from 2) whose recursive definition is similar to the Fibonacci numbers. Further understanding this polynomial seems to be a promising route to new formula lower bounds.

Our methods are rooted in the study of ABPs of small constant width. In 1992 Ben-Or and Cleve showed that formula size is polynomially equivalent to width-3 ABP size. We extend their result (in characteristic different from 2) by showing that approximate formula size is polynomially equivalent to approximate width-2 ABP size. This is surprising because in 2011 Allender and Wang gave explicit polynomials that cannot be computed by width-2 ABPs at all! The details of our construction lead to the aforementioned characterization of $\overline{\mathbf{VP}_{\mathrm{e}}}$.

As a natural continuation of this work we prove that the class $\mathbf{VNP}$ can be described as the class of families that admit a hypercube summation of polynomially bounded dimension over a product of polynomially many affine linear forms. This gives the first separations of algebraic complexity classes from their nondeterministic analogs.

1998 ACM Subject Classification F.1.3 Complexity Measures and Classes

Keywords and phrases algebraic branching programs, algebraic complexity theory, border complexity, formula size, iterated matrix multiplication

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.20

1 Introduction

Let $\mathbf{VP}_{\mathrm{e}}$ denote the class of families of polynomials with polynomially bounded formula size and let $\mathbf{VP}_{\mathrm{s}}$ denote the class of families of polynomials that can be written as determinants of matrices of polynomially bounded size whose entries are affine linear forms. In 1979 Valiant [53] proved his famous result $\mathbf{VP}_{\mathrm{e}} \subseteq \mathbf{VP}_{\mathrm{s}}$. The question whether this inclusion is

* This work was partially supported by NWO (617.023.116).

32nd Computational Complexity Conference (CCC 2017).
Editor: Ryan O'Donnell; Article No. 20; pp. 20:1–20:31

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

strict is a long-standing open question in algebraic complexity theory: Can the determinant polynomial $\det_n := \sum_{\sigma \in S_n} \operatorname{sgn}(\sigma) \prod_{i=1}^{n} x_{i,\sigma(i)}$ be computed by formulas of polynomially bounded size? Motivated by this question we study the class $\overline{\mathbf{VP}_\mathrm{e}}$ of families of polynomials that can be approximated arbitrarily closely by families in $\mathbf{VP}_\mathrm{e}$ (see Section 2 for a formal definition). We present a simple description of the closure $\overline{\mathbf{VP}_\mathrm{e}}$ and of a $\overline{\mathbf{VP}_\mathrm{e}}$-complete polynomial whose recursive definition is similar to the Fibonacci numbers, given the characteristic is not 2, see Theorem 3.11.

In algebraic complexity theory, the way of showing a complexity lower bound for a problem $f \in V$ for some $\mathbb{F}$-vector space V most often goes by (implicitly or explicitly) finding a function $\mathcal{F} : V \to \mathbb{F}$ that is zero on all problems of low complexity while at the same time $\mathcal{F}(f) \neq 0$. Grochow [20] gives a long list (e.g., [41, 44, 34, 23, 32, 13]) of settings where complexity lower bounds are obtained in this way. Moreover, he points out that over the complex numbers these functions $\mathcal{F}$ can be assumed to be continuous (and even to be so-called highest-weight vector polynomials). If $\mathbf{C}$ and $\mathbf{D}$ are algebraic complexity classes with $\mathbf{C} \subseteq \mathbf{D}$ (for example, $\mathbf{C} = \mathbf{VP}_\mathrm{e}$ and $\mathbf{D} = \mathbf{VP}_\mathrm{s}$), then any separation of algebraic complexity classes $\mathbf{C} \neq \mathbf{D}$ in this continuous manner would automatically imply the stronger statement $\mathbf{D} \not\subseteq \overline{\mathbf{C}}$. It is therefore natural to try to prove the separation $\mathbf{VP}_\mathrm{s} \not\subseteq \overline{\mathbf{VP}_\mathrm{e}}$ instead of the slightly weaker $\mathbf{VP}_\mathrm{e} \neq \mathbf{VP}_\mathrm{s}$, which provides further motivation for studying $\overline{\mathbf{VP}_\mathrm{e}}$. This is exactly analogous to Mulmuley and Sohoni's geometric complexity approach (see e.g. [38, 39] and the exposition [15, Sec. 9]) where one tries to prove the separation $\mathbf{VNP} \not\subseteq \overline{\mathbf{VP}_\mathrm{s}}$ to attack Valiant's famous $\mathbf{VP}_\mathrm{s} \neq \mathbf{VNP}$ conjecture [53]. Here $\mathbf{VNP}$ is the class of p-definable families, see Section 2 for a precise definition.

The generalized Fibonacci polynomial

We prove that the *generalized Fibonacci polynomial* F_n is $\overline{\mathbf{VP}_\mathrm{e}}$-complete under p-degenerations, where F_n is defined via $F_0 := 1$, $F_1 := x_1$, $F_n := x_n F_{n-1} + F_{n-2}$, see Section 3. This means that every family (f_n) in $\overline{\mathbf{VP}_\mathrm{e}}$ can be obtained as the limit of a sequence $f_n = \lim_{j\to\infty} F_{t(n)}(\ell_1(j), \ldots, \ell_{t(n)}(j))$, where each $\ell_i(j)$ is a variable or constant and $t(n)$ is a polynomially bounded function. This is arguably the simplest $\overline{\mathbf{VP}_\mathrm{e}}$-complete polynomial known today. Prior to our work the simplest $\overline{\mathbf{VP}_\mathrm{e}}$-complete (and $\mathbf{VP}_\mathrm{e}$-complete) polynomial was the iterated 3×3 matrix multiplication polynomial [6]. This immediately motivates the definition of *border Fibonacci complexity* $\underline{L}_{\mathrm{Fib}}(f)$ of a polynomial f, which is the smallest number m such that f can be obtained as $\lim_{j\to\infty}(F_m(\ell_1(j), \ldots, \ell_m(j)))_j$. To make the situation more geometric we allow the $\ell_i(j)$ to be arbitrary affine linear forms. Our results show that border Fibonacci complexity is polynomially equivalent to border formula size. This insight is quite striking because a result of Allender and Wang [2] implies that the Fibonacci complexity *without allowing approximations* can be infinite!

A promising path towards proving formula lower bounds, for example for the determinant or the permanent, is to apply to our setting the following standard geometric ideas. If we take our field to be the complex numbers and fix the number of variables n and the degree d, then the set of homogeneous degree d polynomials $\mathbb{C}[x_1, \ldots, x_n]_d$ contains the set

$$X_m := \{f \in \mathbb{C}[x_1, \ldots, x_n]_d \mid \underline{L}_{\mathrm{Fib}}(f) \leq m\}$$

as an affine subvariety (X_m is the closure of the set of affine projections of F_m intersected with $\mathbb{C}[x_1, \ldots, x_n]_d$). Moreover, since we allowed the $\ell_i(j)$ to be affine linear forms, the group $\mathrm{GL}(\mathbb{C}^n)$ acts canonically on X_m, making X_m an affine $\mathrm{GL}(\mathbb{C}^n)$-variety. If we find a polynomial $\mathcal{F}$ that vanishes identically on X_m, then a nonzero evaluation $\mathcal{F}(f) \neq 0$ implies that $\underline{L}_{\mathrm{Fib}}(f) > m$. This approach looks feasible given the very simple structure of the

generalized Fibonacci polynomial. This is emphasized by the fact that the action of $\mathrm{GL}(\mathbb{C}^n)$ puts a lot of structure on the coordinate ring of X_m, see for example [12, 5, 34, 13, 26, 22, 42] where the action of the general linear group on the coordinate ring of a variety is used to classify some of its defining equations.

1.1 Main Results

Algebraic Branching Programs (ABPs) of width 2

Our main objects of study are the following classes of families of polynomials: the class of families of polynomials with polynomially bounded formula size $\mathbf{VP}_\mathrm{e}$ (fan-in 2 arithmetic formulas that use additions and multiplications as their operations), its closure $\overline{\mathbf{VP}_\mathrm{e}}$, and the nondeterministic variant $\mathbf{VNP}$. We do so by studying algebraic branching programs of small width. These are defined as follows. An *algebraic branching program* (ABP) is a directed acyclic graph with a source vertex s and a sink vertex t that has affine linear forms over the base field $\mathbb{F}$ as edge labels. Moreover, we require that each vertex is labeled with an integer (its *layer*) and that edges in the ABP only point from vertices in layer i to vertices in layer $i+1$. The *width* of an ABP is the cardinality of its largest layer. The *size* of an ABP is the number of its vertices. The *value* of an ABP is the sum of the values of all s-t-paths, where the value of an s-t-path is the product of its edge labels. We say that an ABP *computes* its value. The class $\mathbf{VP}_\mathrm{s}$ coincides with the class of families of polynomials that can be computed by ABPs of polynomially bounded size, see e.g. [47].

For this paper we introduce the class $\mathbf{VP}_k$, $k \in \mathbb{N}$, which is defined as the class of families of polynomials computable by width-k ABPs of polynomially bounded size. It is well-known that $\mathbf{VP}_k \subseteq \mathbf{VP}_\mathrm{e}$ for every $k \geq 1$ (see Proposition 7.1). In 1992, Ben-Or and Cleve [6] showed that $\mathbf{VP}_k = \mathbf{VP}_\mathrm{e}$ for all $k \geq 3$ (we review the proof, see Theorem 6.1). In 2011 Allender and Wang [2] showed that width-2 ABPs cannot compute every polynomial, so in particular we have a strict inclusion $\mathbf{VP}_2 \subsetneq \mathbf{VP}_3$. Let the characteristic of the base field $\mathbb{F}$ be different from 2. Our first main result (Theorem 3.1 and Corollary 3.8) is that the closure of $\mathbf{VP}_2$ and the closure of $\mathbf{VP}_\mathrm{e}$ are equal,

$$\overline{\mathbf{VP}_2} = \overline{\mathbf{VP}_\mathrm{e}}. \tag{1}$$

Interestingly, as a direct corollary of (1) and the result of Allender and Wang, the inclusion $\mathbf{VP}_2 \subsetneq \overline{\mathbf{VP}_2}$ is strict. It is easy to see that $\mathbf{VP}_1$ equals $\overline{\mathbf{VP}_1}$ (Proposition 5.10), so $\mathbf{VP}_1$ and $\mathbf{VP}_2$ are examples of quite similar algebraic complexity classes that behave differently under closure. Most importantly, from the proof of (1) we obtain our results about the generalized Fibonacci polynomial that we mentioned before.

VNP via affine linear forms

We define the classes $\mathbf{VNP}_\mathrm{e}$ and $\mathbf{VNP}$ in the natural way. In 1980, Valiant [54] showed that $\mathbf{VNP}_\mathrm{e} = \mathbf{VNP}$ and in this paper we will always view $\mathbf{VNP}$ as the nondeterministic analog of $\mathbf{VP}_\mathrm{e}$. To $\mathbf{VP}_1$ and $\mathbf{VP}_2$ we similarly associate nondeterministic analogs $\mathbf{VNP}_1$ and $\mathbf{VNP}_2$ (see Section 2). Using interpolation techniques it is possible to deduce $\mathbf{VNP}_2 = \mathbf{VNP}$ from (1), provided the field is infinite. Using more sophisticated techniques we strengthen this result to get our second main result (Theorem 4.2):

$$\mathbf{VNP}_1 = \mathbf{VNP}. \tag{2}$$

That is, a family (f_n) is contained in $\mathbf{VNP}$ iff f_n can be written as a hypercube summation of polynomially bounded dimension over a product of polynomially many affine linear

forms. Using (2) it is then easy to verify that $\mathbf{VP}_1 \subsetneq \mathbf{VNP}_1$ and using [2] yields $\mathbf{VP}_2 \subsetneq \mathbf{VNP}_2$, which separates complexity classes from their nondeterministic analogs. Interestingly $\mathbf{VNP}_1 \subsetneq \mathbf{VNP}$ over the field with 2 elements, see Section 9.

Restricted ABP edge labels

Several more results on small-width ABPs, approximation closures, and hypercube summations are proved throughout this paper. For example, in Section 5 we investigate the subtleties of what happens if we restrict the ABP edge labels to simple affine linear forms, or to variables and constants. The precise relations between complexity classes that we obtain are listed in Figure A in Appendix A. As another example, we strengthen (2) as follows (Theorem 6.2): A family (f_n) is contained in **VNP** iff f_n can be written as a hypercube summation of polynomially bounded dimension over a product of polynomially many affine linear forms that use *at most two variables* each.

1.2 Related work

In the boolean setting as well as in the algebraic setting finding lower bounds for the formula size of explicit problems is considered a major open problem. For the boolean setting we refer the reader to the line of papers [49, 4, 28, 43, 25, 50], which results in an explicit function with formula size $\Omega(n^3/(\log^2 n \log\log n))$.

In the algebraic setting the smallest formula for the determinant has size $\mathcal{O}(n^{\log n})$, which can be deduced from e.g. [27]. The best known lower bound on the formula size of $\det_n$ is $\Omega(n^3)$ by [29]. That paper also gives a quadratic lower bound for an explicit polynomial (note that the lower bound for the determinant is not quadratic in the number of variables).

Toda [52] proved that several definitions for the class $\mathbf{VP}_\mathrm{s}$ are equivalent, see also [36]. In particular $\mathbf{VP}_\mathrm{s}$ is the class of polynomials that can be written as determinants of matrices of polynomially bounded size whose entries are affine linear forms. Due to its pure mathematical formulation, lower bounds for this *determinantal complexity* attracted the attention of geometers [37, 32, 3]. Moreover, Mulmuley and Sohoni's geometric complexity approach [38, 39] is also mainly focused on lower bounds for the determinantal complexity and the symmetries of the determinant polynomial play a key role in their work. Recently [14] showed that it is not possible to prove superpolynomial lower bounds on the determinantal complexity using only information about the occurrences/non-occurrences of irreducible representations in the coordinate rings of the orbit closures of the determinant and the (padded) permanent. This disproves a major conjecture in geometric complexity theory. The proof in [14] is fairly general and also holds for lower bounds on the formula size. Only very recently the formula size analog to determinantal complexity, the *iterated matrix multiplication complexity* was studied from a geometric perspective [19].

There is a large number of publications on lower bounds for constant *depth* circuits and formulas (with superconstant fan-in), see e.g. [1, 30, 24, 51], which recently led to the celebrated result [23] that the permanent does not admit size $2^{o(\sqrt{m})}$ homogeneous $\Sigma\Pi\Sigma\Pi$ circuits in which the bottom fan-in is bounded by $\sqrt{m}$. In the light of the previous depth-reduction results this seemed very close to separating **VP** from **VNP**. Several very recent results [16, 18] indicate that new ideas are needed to separate **VP** from **VNP**.

Ben-Or and Cleve [6] proved that a family of polynomials has polynomially bounded formula size if and only if it is computable by width-3 ABPs of polynomial size. An excellent exposition on the history of small-width computation can be found in [2], along with an explicit polynomial that cannot be computed by width-2 ABPs: $x_1x_2 + x_3x_4 + \cdots + x_{15}x_{16}$.

Saha, Saptharishi and Saxena [46, Cor. 14] showed that $x_1x_2 + x_3x_4 + x_5x_6$ cannot be computed by width-2 ABPs that correspond to the iterated matrix multiplication of upper triangular matrices.

Bürgisser [10] studied approximations in the model of general algebraic circuits, finding general upper bounds on the error degree. For most specific algebraic complexity classes $\mathbf{C}$ the relation between $\mathbf{C}$ and $\overline{\mathbf{C}}$ has not been an active object of study. As pointed out recently by Forbes [17], Nisan's result [40] implies that $\mathbf{C} = \overline{\mathbf{C}}$ for $\mathbf{C}$ being the class of size-k algebraic branching programs on noncommuting variables. Recently, a structured study of $\overline{\mathbf{VP}}$ and $\overline{\mathbf{VP}_\mathrm{s}}$ has been started, see [21]. By far the most work in lower bounds for topological approximation algorithms has been done in the area of bilinear complexity, dating back to [7, 48, 35] and more recently [31, 34, 26, 55, 33], to list a few.

1.3 Paper outline

In Section 2 we introduce in more detail the approximation closure and the nondeterminism closure of a complexity class. In Section 3 we prove the first main result: border formula size is polynomially equivalent to border width-2 ABP size and the generalized Fibonacci polynomial is $\overline{\mathbf{VP}_\mathrm{e}}$-complete under p-degenerations. In Section 4 we prove the second main result: a new description of $\mathbf{VNP}$ as the nondeterminism closure of families that have polynomial-size width-1 ABPs. The later sections contain details on how to strengthen the result from Section 4 and results on the power of ABPs with restricted edge labels.

2 Nondeterminism and approximation closure

In this section we introduce the approximation closure and the nondeterminism analog of a class. A *family* is a sequence of polynomials $(f_n)_{n\in\mathbb{N}}$. A *class* is a set of families and will be written in boldface, $\mathbf{C}$. For an introduction to the algebraic complexity classes $\mathbf{VP}_\mathrm{e}$, $\mathbf{VP}$, and $\mathbf{VNP}$ we refer the reader to [11]. We denote by $\mathrm{poly}(n)$ the set of polynomially bounded functions $\mathbb{N} \to \mathbb{N}$. We define the norm of a complex multivariate polynomial as the sum of the absolute values of its coefficients. This defines a topology on the polynomial ring $\mathbb{C}[x_1, \ldots, x_m]$. Given a complexity measure L, say ABP size or formula size, there is a natural notion of approximate complexity that is called *border complexity*. Namely, a polynomial $f \in \mathbb{C}[\mathbf{x}]$ has *border complexity* $\underline{L}^{\mathrm{top}}$ at most c if there is a sequence of polynomials $g_1, g_2, \ldots$ in $\mathbb{C}[\mathbf{x}]$ converging to f such that each g_i satisfies $L(g_i) \leq c$. It turns out that for reasonable classes over the field of complex numbers $\mathbb{C}$, this *topological* notion of approximation is equivalent to what we call *algebraic* approximation (see e.g. [10]). Namely, a polynomial $f \in \mathbb{C}[\mathbf{x}]$ satisfies $\underline{L}(f)^{\mathrm{alg}} \leq c$ iff there are polynomials $f_1, \ldots, f_e \in \mathbb{C}[\mathbf{x}]$ such that the polynomial

$$h := f + \varepsilon f_1 + \varepsilon^2 f_2 + \cdots + \varepsilon^e f_e \in \mathbb{C}[\varepsilon, \mathbf{x}]$$

has complexity $L_{\mathbb{C}(\varepsilon)}(h) \leq c$, where ε is a formal variable and $L_{\mathbb{C}(\varepsilon)}(h)$ denotes the complexity of h over the field extension $\mathbb{C}(\varepsilon)$. This algebraic notion of approximation makes sense over any base field and we will use it in the statements and proofs of this paper.

▶ **Definition 2.1.** Let $\mathbf{C}(\mathbb{F})$ be a class over the field $\mathbb{F}$. We define the *approximation closure* $\overline{\mathbf{C}}(\mathbb{F})$ as follows: a family (f_n) over $\mathbb{F}$ is in $\overline{\mathbf{C}}(\mathbb{F})$ if there are polynomials $f_{n;i}(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$ and a function $e : \mathbb{N} \to \mathbb{N}$ such that the family (g_n) defined by

$$g_n(\mathbf{x}) := f_n(\mathbf{x}) + \varepsilon f_{n;1}(\mathbf{x}) + \varepsilon^2 f_{n;2}(\mathbf{x}) + \cdots + \varepsilon^{e(n)} f_{n;e(n)}(\mathbf{x})$$

is in $\mathbf{C}(\mathbb{F}(\varepsilon))$. We define the *poly-approximation closure* $\overline{\mathbf{C}}^{\mathrm{poly}}(\mathbb{F})$ similarly, but with the additional requirement that $e(n) \in \mathrm{poly}(n)$. We call $e(n)$ the *error degree*.

Interestingly, for subtle reasons, taking the approximation closure $\mathbf{C} \mapsto \overline{\mathbf{C}}$ is *not* idempotent in general and hence not a closure operator, but for reasonable classes (like $\mathbf{VP}_k$, $\mathbf{VP}_e$, and $\mathbf{VP}$) it is.

One can think of $\mathbf{VNP}$ as a "nondeterminism closure" of $\mathbf{VP}$. We want to use the nondeterminism closure for general classes.

▶ **Definition 2.2.** Let $\mathbf{C}$ be a class. The class $\mathrm{N}(\mathbf{C})$ consists of families (f_n) with the following property: there is a family $(g_n) \in \mathbf{C}$ and $p(n), q(n) \in \mathrm{poly}(n)$ such that

$$f_n(\mathbf{x}) = \sum_{\mathbf{b} \in \{0,1\}^{p(n)}} g_{q(n)}(\mathbf{b}, \mathbf{x}),$$

where $\mathbf{x}$ and $\mathbf{b}$ denote sequences of variables $x_1, x_2, \ldots$ and $b_1, b_2, \ldots, b_{p(n)}$. We will sometimes say that $f(\mathbf{x})$ is a *hypercube sum over* g and that $b_1, b_2, \ldots, b_{p(n)}$ are the *hypercube variables*. For any s, t, we will use the standard notation $\mathbf{VNP}_s^t$ to denote $\mathrm{N}(\mathbf{VP}_s^t)$, where the superscript t will become relevant in Section 5. We remark that the map $\mathbf{C} \mapsto \mathrm{N}(\mathbf{C})$ trivially satisfies all properties of being a closure operator.

3 Approximate width-2 ABPs and formula size

As mentioned in the introduction, Allender and Wang [2] showed that there exist polynomials that cannot be computed by any width-2 ABP, for example the polynomial $x_1x_2 + x_3x_4 + \cdots + x_{15}x_{16}$. Therefore, we have a separation $\mathbf{VP}_2 \subsetneq \mathbf{VP}_3 = \mathbf{VP}_e$. We show that allowing approximation changes the situation completely: every polynomial can be approximated by a width-2 ABP. In fact, every polynomial can be approximated by a width-2 ABP of size polynomial in the formula size, and with error degree polynomial in the formula size. This is the main result of this section.

▶ **Theorem 3.1.** $\mathbf{VP}_e \subseteq \overline{\mathbf{VP}_2}^{\mathrm{poly}}$ *when* $\mathrm{char}(\mathbb{F}) \neq 2$.

We leave as an open question what happens in characteristic 2.

In order to understand the following proofs and the corresponding figures it is advisable to recall that an ABP corresponds naturally to an iterated product of matrices if we number the vertices in each layer consecutively, starting with 1. Namely, consider two consecutive layers i and $i+1$ and let M_i be the matrix whose entry at position (v, w) is the label of the edge from vertex v in layer i to vertex w in layer $i+1$ (or 0 if there is no edge between these vertices). Then the ABP's value equals the product $M_k \cdots M_2 M_1$.

For a polynomial f over $\mathbb{F}(\varepsilon)$ define the matrix $Q(f) \coloneqq \left(\begin{smallmatrix} f & 1 \\ 1 & 0 \end{smallmatrix}\right)$. A *parametrized affine linear form* is an affine linear form over the field $\mathbb{F}(\varepsilon)$. A *primitive Q-matrix* is any matrix $Q(\ell)$, where ℓ is a parametrized linear form. For a 2×2 matrix M with entries in $\mathbb{F}(\varepsilon)[\mathbf{x}]$, we use the shorthand notation $M + \mathcal{O}(\varepsilon^k)$ for $M + \left(\begin{smallmatrix} \mathcal{O}(\varepsilon^k) & \mathcal{O}(\varepsilon^k) \\ \mathcal{O}(\varepsilon^k) & \mathcal{O}(\varepsilon^k) \end{smallmatrix}\right)$, where $\mathcal{O}(\varepsilon^k)$ denotes the set $\varepsilon^k \mathbb{F}[\varepsilon, \mathbf{x}]$. As a product of matrices, the ABP construction in our proof of Theorem 3.1 will be of the form $\left(\begin{smallmatrix} 1 & 0 \end{smallmatrix}\right) M_\ell \cdots M_2 M_1 \left(\begin{smallmatrix} 1 \\ 0 \end{smallmatrix}\right)$ where the M_i are primitive Q-matrices $Q(f)$ for which f is either a constant from $\mathbb{F}(\varepsilon)$ or a variable. We are thus proving a slightly stronger statement than the statement of Theorem 3.1.

▶ **Lemma 3.2** (Addition). *Let $k \geq 1$. Let $f, g \in \mathbb{F}[\mathbf{x}]$ be polynomials such that some $F \in Q(f) + \mathcal{O}(\varepsilon^k)$ and some $G \in Q(g) + \mathcal{O}(\varepsilon^k)$ can be written as a product of n and m primitive Q-matrices, respectively. Then some matrix $H \in Q(f+g) + \mathcal{O}(\varepsilon^k)$ can be written as the product of $n + m + 1$ primitive Q-matrices. Moreover, if the error degrees in F, G are e_f, e_g, respectively, then the error degree of H is at most $e_f + e_g$.*

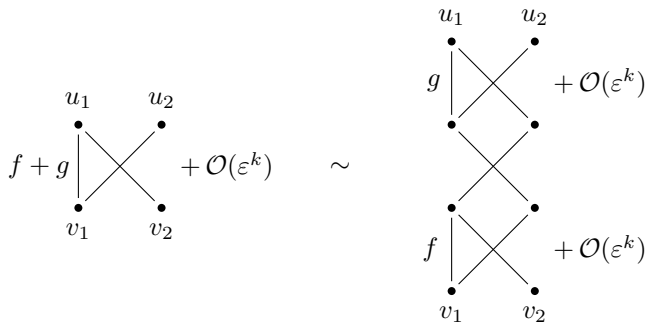


Figure 1 Addition construction for Lemma 3.2.

Proof. Note that $(Q(f)+\mathcal{O}(\varepsilon^k))\cdot Q(0)\cdot(Q(g)+\mathcal{O}(\varepsilon^k)) = Q(f+g)+\mathcal{O}(\varepsilon^k)$, so we have $H \coloneqq F\cdot Q(0)\cdot G \in Q(f+g)+\mathcal{O}(\varepsilon^k)$. Moreover, the largest power of ε occurring in H is $\varepsilon^{e_f+e_g}$. See Fig. 1. ◀

▶ **Lemma 3.3** (Squaring). *Let $f\in\mathbb{F}[\mathbf{x}]$ be a polynomial such that some $F\in Q(f)+\mathcal{O}(\varepsilon^3)$ can be written as the product of n primitive Q-matrices. Then some matrix $H\in Q(f^2)+\mathcal{O}(\varepsilon)$ and some matrix $H'\in Q(-f^2)+\mathcal{O}(\varepsilon)$ can be written as the product of $2n+11$ primitive Q-matrices. Moreover, if the error degree in F is e_f then the error degree of H and H' is at most $2\cdot e_f+4$.*

Proof. We set

$$A \coloneqq \begin{pmatrix} -\varepsilon^{-1} & 0 \\ 0 & \varepsilon \end{pmatrix} = Q(-\varepsilon^{-1})\cdot Q(\varepsilon)\cdot Q(-\varepsilon^{-1}),$$
$$B \coloneqq \begin{pmatrix} \varepsilon^2 & 1 \\ -1 & 0 \end{pmatrix} = Q(1)\cdot Q(-1)\cdot Q(1)\cdot Q(\varepsilon^2),$$
$$C \coloneqq \begin{pmatrix} \varepsilon^{-1} & 0 \\ 0 & \varepsilon \end{pmatrix} = Q(-\varepsilon^{-1})\cdot Q(\varepsilon-1)\cdot Q(1)\cdot Q(\varepsilon^{-1}-1).$$

Then one can check that

$$H \;\coloneqq\; A\cdot F\cdot B\cdot F\cdot C \;\in\; A\cdot(Q(f)+\mathcal{O}(\varepsilon^3))\cdot B\cdot(Q(f)+\mathcal{O}(\varepsilon^3))\cdot C \;\in\; Q(-f^2)+\mathcal{O}(\varepsilon).$$

To obtain $H'\in Q(f^2)+\mathcal{O}(\varepsilon)$, we replace B by

$$B' \coloneqq \begin{pmatrix} -\varepsilon^2 & 1 \\ -1 & 0 \end{pmatrix} = Q(1)\cdot Q(-1)\cdot Q(1)\cdot Q(-\varepsilon^2).$$

One checks that the highest power of ε appearing in H and H' is at most $2\cdot e_f+4$. See Fig. 2 and Fig. 3 for a pictorial description. ◀

▶ **Lemma 3.4** (Multiplication). *Let $f,g\in\mathbb{F}[\mathbf{x}]$ be polynomials such that some $F\in Q(f/2)+\mathcal{O}(\varepsilon^3)$ and some $G\in Q(g)+\mathcal{O}(\varepsilon^3)$ can be written as the product of n and m primitive Q-matrices respectively. Then some $H\in Q(f\cdot g)+\mathcal{O}(\varepsilon)$ can be written as the product of $4n+4m+37$ primitive Q-matrices. Moreover, if the error degrees in F, G are e_f, e_g, respectively, then the error degree of H is at most $4\cdot e_f+4\cdot e_g+12$.*

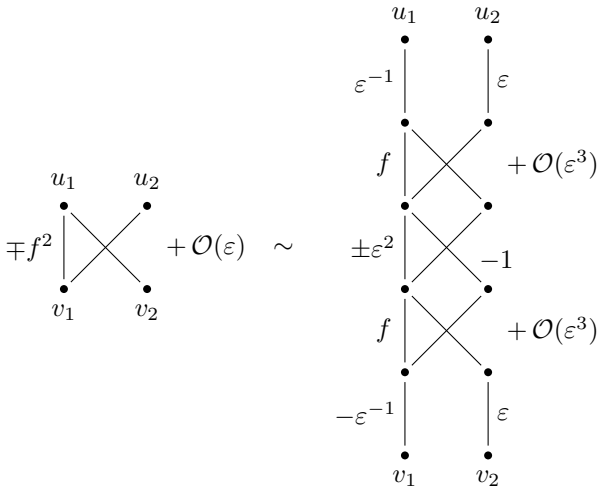


Figure 2 Squaring construction for Lemma 3.3.

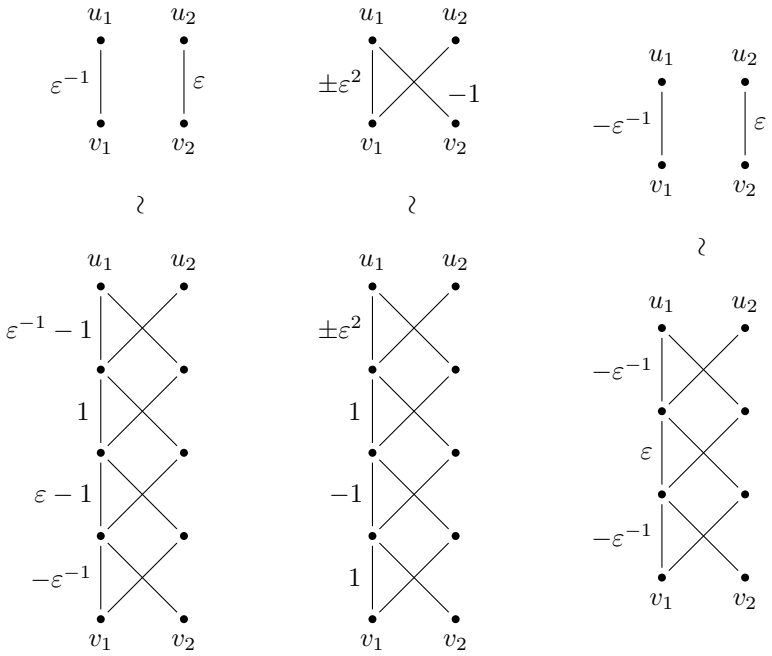


Figure 3 Squaring construction subroutines for C, B, and A for Lemma 3.3.

Proof. We make use of the identity $(-(f/2)^2) + (-g^2) + (f/2+g)^2 = f \cdot g$. By the addition lemma (Lemma 3.2), $(f/2+g)+\mathcal{O}(\varepsilon^3)$ can be written as the product of $n+m+1$ primitive Q-matrices with error degree at most e_f+e_g. By the squaring lemma (Lemma 3.3), $Q(-(f/2)^2)+\mathcal{O}(\varepsilon)$, $Q(-g^2)+\mathcal{O}(\varepsilon)$, and $Q((f/2+g)^2)+\mathcal{O}(\varepsilon)$ can be written as the product of $2n+11$, $2m+11$, and $2(n+m+1)+11$ primitive Q-matrices, respectively. The corresponding error degrees are at most $2\cdot e_f+4$, $2\cdot e_g+4$, and $2(e_f+e_g)+4$. Finally, by the addition lemma again, $Q(f\cdot g)+\mathcal{O}(\varepsilon) = Q(-(f/2)^2+(-g^2)+(f/2+g)^2)+\mathcal{O}(\varepsilon)$ can be written as the product of $(2n+11)+1+(2m+11)+1+(2(n+m+1)+11) = 4n+4m+37$ primitive Q-matrices. The corresponding error degree is at most $(2\cdot e_f+4)+(2\cdot e_g+4)+(2(e_f+e_g)+4) = 4\cdot e_f+4\cdot e_g+12$. See Fig. 4 for a pictorial description. ◀

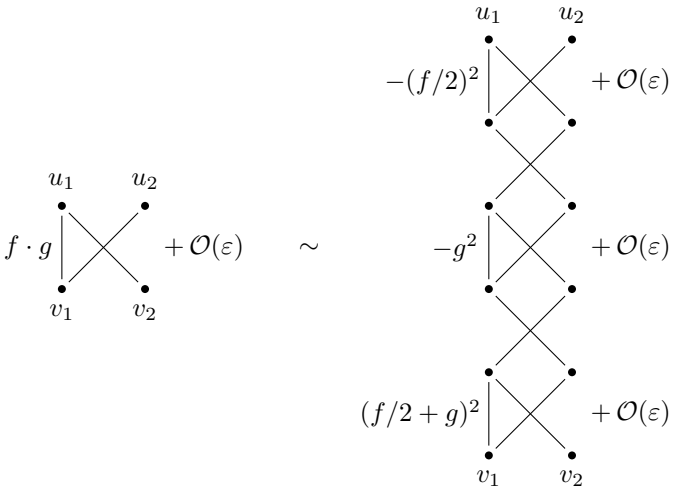


Figure 4 Multiplication construction for Lemma 3.4.

▶ **Proposition 3.5.** *Let f be a polynomial computed by a formula of depth d. For every constant $\alpha \in \mathbb{F}$, some matrix in $F \in Q(\alpha f) + \mathcal{O}(\varepsilon)$ can be written as a product of at most $45 \cdot 9^d$ primitive Q-matrices. Moreover, F has error degree at most $12 \cdot 25^d$.*

Proof. The proof is by induction on d. For $d = 0$, that is, f is a constant $\beta \in \mathbb{F}$ or a variable x, note that $Q(f)$ can be written directly as a primitive Q-matrix (with error degree 0). Since also $Q(\alpha/2)$ can be written directly (also with error degree 0), we can use the multiplication lemma (Lemma 3.4), to write $Q(\alpha f) + \mathcal{O}(\varepsilon)$ as a product of $4+4+37 = 45$ primitive Q-matrices (with error degree at most 12).

For $d \geq 1$, fix a constant α. We know that either $f = g + h$ or $f = g \cdot h$ with formulas g, h of depth $< d$. By the induction hypothesis, for any constant β, γ, we can write $Q(\beta g) + \mathcal{O}(\varepsilon)$ and $Q(\gamma h) + \mathcal{O}(\varepsilon)$ as a product of $n_g, n_h \leq 45 \cdot 9^{d-1}$ primitive Q-matrices, with error degrees $e_g, e_h \leq 12 \cdot 25^{d-1}$.

Case $f = g + h$. We set $\beta = \gamma = \alpha$ and use the addition lemma (Lemma 3.2) to obtain $Q(\alpha f) + \mathcal{O}(\varepsilon) = Q(\alpha g + \alpha h) + \mathcal{O}(\varepsilon)$ as a product of $n_g + n_h + 1 \leq 2 \cdot 45 \cdot 9^{d-1} + 1 \leq 45 \cdot 9^d$ primitive Q-matrices, with error degree at most $e_g + e_h \leq 2 \cdot 12 \cdot 25^{d-1} \leq 12 \cdot 25^d$.

Case $f = g \cdot h$. By replacing ε by ε^3 in all primitive Q-matrices, we obtain matrices in $Q(\beta g) + \mathcal{O}(\varepsilon^3)$ and $Q(\gamma h) + \mathcal{O}(\varepsilon^3)$ as a product of n_g and n_h primitive Q-matrices with error degree at most $3 \cdot e_g$ and $3 \cdot e_h$ respectively. Now we set $\beta = \alpha/2$ and $\gamma = 1$ and use the multiplication lemma (Lemma 3.4) to obtain $Q(\alpha f) + \mathcal{O}(\varepsilon) = Q((\alpha \cdot g) \cdot h) + \mathcal{O}(\varepsilon)$ as a product of $4n_g + 4n_h + 37 \leq 8 \cdot 45 \cdot 9^{d-1} + 37 \leq 45 \cdot 9^d$ primitive Q-matrices. The error degree is at most $4(3 \cdot e_g) + 4(3 \cdot e_h) + 12 = 12(e_g + e_h + 1) \leq 24 \cdot 12 \cdot 25^{d-1} + 12 \leq 12 \cdot 25^d$. ◀

▶ **Proposition 3.6.** *If $(f_n) \in \mathbf{VP}_{\mathrm{e}}$, then for each n a matrix in $F \in Q(f_n) + \mathcal{O}(\varepsilon)$ can be written as a product of* $\mathrm{poly}(n)$ *many primitive Q-matrices. Moreover, F has error degree at most* $\mathrm{poly}(n)$.

Proof. The construction uses the classical depth-reduction theorem for formulas by Brent [8], for which a modern proof can be found in the survey of Saptharishi [47, Lemma 5.5]: If a family (f_n) has polynomially bounded formula size, then there are formulas computing f_n that have size $\mathrm{poly}(n)$ and depth $\mathcal{O}(\log n)$. Applying Proposition 3.5 now yields the result. ◀

Proof of Theorem 3.1. This follows directly from Proposition 3.6. Namely, let $(f_n) \in \mathbf{VP}_e$. By Proposition 3.6 there is an $F \in Q(f_n) + \mathcal{O}(\varepsilon)$ which is a product of polynomially many primitive Q-matrices such that F has polynomially bounded error degree. The width-2 ABP computing $f_n + \mathcal{O}(\varepsilon)$ is given by $\left(\begin{smallmatrix}1 & 0\end{smallmatrix}\right) F \left(\begin{smallmatrix}1\\0\end{smallmatrix}\right)$. ◀

▶ **Example 3.7.** Following the construction in Theorem 3.1 we get the following ABP for approximating the polynomial $x_1x_2 + x_3x_4 + \cdots + x_{15}x_{16}$, which cannot be computed by any width-2 ABP. Let

$$F(x_1,x_2) = \begin{pmatrix} \frac{1}{\varepsilon} - \frac{\varepsilon x_1}{2} & -\frac{x_1}{2\varepsilon} \\ \varepsilon^3 & \varepsilon \end{pmatrix} \begin{pmatrix} \frac{1}{2}(x_1 - 2x_2)\varepsilon^2 + 1 & \frac{1}{2}(x_1 - 2x_2) \\ \varepsilon^2 & 1 \end{pmatrix} \cdot \begin{pmatrix} \frac{x_1\varepsilon^2}{2} + 1 & -\frac{x_1}{2} \\ -\varepsilon^2 & 1 \end{pmatrix} \begin{pmatrix} \frac{x_1+2x_2}{2\varepsilon} & \varepsilon \\ \varepsilon^{-1} & 0 \end{pmatrix}.$$

Then

$$F(x_1,x_2) = \begin{pmatrix} x_1x_2 & 1 \\ 1 & 0 \end{pmatrix} + \mathcal{O}(\varepsilon).$$

Using the addition lemma Lemma 3.2 we get

$$\left(\begin{smallmatrix}1 & 0\end{smallmatrix}\right) F(x_1,x_2) \left(\begin{smallmatrix}0 & 1\\1 & 0\end{smallmatrix}\right) F(x_3,x_4) \cdots \left(\begin{smallmatrix}0 & 1\\1 & 0\end{smallmatrix}\right) F(x_{15},x_{16}) \left(\begin{smallmatrix}1\\0\end{smallmatrix}\right) = x_1x_2 + x_3x_4 + \cdots + x_{15}x_{16} + \mathcal{O}(\varepsilon).$$

▶ **Corollary 3.8.** $\overline{\mathbf{VP}_2} = \overline{\mathbf{VP}_e}$ *and* $\overline{\mathbf{VP}_2}^{\mathrm{poly}} = \overline{\mathbf{VP}_e}^{\mathrm{poly}}$ *when* $\mathrm{char}(\mathbb{F}) \neq 2$.

Proof. The inclusion $\mathbf{VP}_2 \subseteq \mathbf{VP}_e$ is standard (see Proposition 7.1). Taking closures on both sides, we obtain $\overline{\mathbf{VP}_2} \subseteq \overline{\mathbf{VP}_e}$ and $\overline{\mathbf{VP}_2}^{\mathrm{poly}} \subseteq \overline{\mathbf{VP}_e}^{\mathrm{poly}}$.

On the other hand, when $\mathrm{char}(\mathbb{F}) \neq 2$, we have the inclusion $\mathbf{VP}_e \subseteq \overline{\mathbf{VP}_2}^{\mathrm{poly}}$ (Theorem 3.1). By taking closures this implies $\overline{\mathbf{VP}_e} \subseteq \overline{\mathbf{VP}_2}$ and $\overline{\mathbf{VP}_e}^{\mathrm{poly}} \subseteq \overline{\mathbf{VP}_2}^{\mathrm{poly}}$. ◀

▶ **Corollary 3.9.** $\overline{\mathbf{VP}_2}^{\mathrm{poly}} = \mathbf{VP}_e$ *when* $\mathrm{char}(\mathbb{F}) \neq 2$ *and* $\mathbb{F}$ *is infinite.*

Proof. By Corollary 3.8 we have $\overline{\mathbf{VP}_2}^{\mathrm{poly}} = \overline{\mathbf{VP}_e}^{\mathrm{poly}}$. It remains to show the equality $\overline{\mathbf{VP}_e}^{\mathrm{poly}} = \mathbf{VP}_e$. We give a proof of this via a standard interpolation argument in Section 8. ◀

As a consequence of Proposition 3.5, we obtain a new description of $\overline{\mathbf{VP}_e}$ as follows. We define the *generalized Fibonacci polynomial* $F_n(x_1,\ldots,x_n)$ by $F_0 := 1$, $F_1 := x_1$, and $F_n := x_nF_{n-1} + F_{n-2}$ for all $n \geq 2$. The name comes from the fact that $F_n(1,1,\ldots,1)$ is the nth Fibonacci number and $F_n(x,x,\ldots,x)$ is the nth Fibonacci polynomial. Another description of the polynomial F_n is that it is the upper left entry of a product of Q-matrices $Q(x_i)$, that is, $F_n(x_1,\ldots,x_n) = (Q(x_n)Q(x_{n-1})\cdots Q(x_1))_{1,1}$.

▶ **Definition 3.10.** A polynomial f is a *projection* of F_m if there exist affine linear forms $\ell_1,\ldots,\ell_m$ such that $f = F_m(\ell_1,\ldots,\ell_m)$. The smallest m such that f is a projection of F_m we call the *Fibonacci complexity of* f. A polynomial is a *degeneration of* F_m if there exist parametrized affine linear forms $\ell_1(\varepsilon),\ldots,\ell_m(\varepsilon)$ such that $f = F_m(\ell_1(\varepsilon),\ldots,\ell_m(\varepsilon))$. The smallest m such that $f + \mathcal{O}(\varepsilon)$ is a degeneration of F_m we call the *border Fibonacci complexity* of f, and is denoted by $\underline{L}_{\mathrm{Fib}}(f)$. A family (h_n) of polynomials is called $\overline{\mathbf{VP}_e}$*-complete under p-degenerations* if $(h_n) \in \overline{\mathbf{VP}_e}$ and for every $(f_n) \in \overline{\mathbf{VP}_e}$ there exists a polynomially bounded function t such that some polynomial in $f_n + \mathcal{O}(\varepsilon)$ is a degeneration of $F_{t(n)}$.

The Fibonacci complexity is not always finite ([2]), but Proposition 3.6 shows that the border Fibonacci complexity $\underline{L}_{\mathrm{Fib}}(f)$ is always finite and that $\overline{\mathbf{VP}_{\mathrm{e}}}$ can be characterized as the class of families with polynomially bounded border Fibonacci complexity:

▶ **Theorem 3.11.** $\overline{\mathbf{VP}_{\mathrm{e}}} = \{(f_n) \mid \underline{L}_{\mathrm{Fib}}(f_n) \in \mathrm{poly}(n)\}$.

Proof. Clearly the right-hand side is contained in the left-hand side. $\mathbf{VP}_{\mathrm{e}}$ is contained in the right-hand side by Proposition 3.6. A moment's thought reveals that the right-hand side is closed under the approximation closure in the sense of Definition 2.1. Thus taking the closure on both sides yields the result. ◀

Theorem 3.11 says that (F_n) is $\overline{\mathbf{VP}_{\mathrm{e}}}$-complete under p-degenerations. From the proof of Proposition 3.5 it follows that also (F_{2n+1}) is $\overline{\mathbf{VP}_{\mathrm{e}}}$-complete under p-degenerations, that is, we only need the F_m with odd index m (this follows from $\det(Q(f)) = -1$).

▶ Remark (Symmetry). Define the polynomial $C_n(x_1, \ldots, x_n)$ as

$$C_n(x_1, \ldots, x_n) \coloneqq \mathrm{trace}(Q(x_n) \cdot Q(x_{n-1}) \cdots Q(x_1)).$$

Since the trace of a matrix product is invariant under cyclic shifts of the matrices, the polynomial $C_n(x_1, \ldots, x_n)$ is invariant under cyclic shifts of the variables $x_1, \ldots, x_n$. Thus C_n can be viewed as a cyclically symmetric version of F_n. (Note that C_n and F_n are also both invariant under reversing the order of the variables $x_1, \ldots, x_n$, that is, mapping $(x_1, \ldots, x_n)$ to $(x_n, \ldots, x_1)$.)

Define the *border cyclic Fibonacci complexity* analogously to the border Fibonacci complexity by replacing F_n by C_n in Definition 3.10. Analogously to Theorem 3.11 we now see that the families (C_n) and (C_{2n+1}) are both $\overline{\mathbf{VP}_{\mathrm{e}}}$-complete under p-degenerations.

▶ Remark (A closed form for F_n and C_n). We describe another way to write F_n and C_n. An *adjacent pair* is a set of two numbers $\{i, i+1\}$ with $1 \leq i < n$. A *supporting set* is the set $\{1, 2, \ldots, n\}$ after removing a disjoint (possibly empty) union of adjacent pairs. For a supporting set S define $x_S \coloneqq \prod_{i \in S} x_i$. Then $F_n(x_1, \ldots, x_n) = \sum_S x_S$, where the sum is over all supporting sets.

We define a *cyclicly adjacent pair* as a set that is either an adjacent pair or the set $\{1, n\}$, if $1 \neq n$. We define a *cyclic supporting set* as the set $\{1, 2, \ldots, n\}$ after removing a disjoint (possibly empty) union of cyclicly adjacent pairs. Then $C_n(x_1, \ldots, x_n) = \sum_S x_S$, where the sum is over all cyclic supporting sets.

▶ Remark (Planarity). We remark that the product of two Q-matrices $Q(x)Q(y)$ can be rewritten as $Q(x)Q(y) = \left(Q(x)\left(\begin{smallmatrix}0&1\\1&0\end{smallmatrix}\right)\right)\left(\left(\begin{smallmatrix}0&1\\1&0\end{smallmatrix}\right)Q(y)\right)$. We also have $Q(x)\left(\begin{smallmatrix}a\\b\end{smallmatrix}\right) = \left(Q(x)\left(\begin{smallmatrix}0&1\\1&0\end{smallmatrix}\right)\right)\left(\begin{smallmatrix}b\\a\end{smallmatrix}\right)$. Consider a width-2 ABP that is a product of primitive Q-matrices,

$$\left(\begin{smallmatrix}a&b\end{smallmatrix}\right)Q(\ell_1)Q(\ell_2)\cdots Q(\ell_k)\left(\begin{smallmatrix}c\\d\end{smallmatrix}\right).$$

By pairing up the ith Q-matrix with the $(i+1)$th Q-matrix for each odd i, and using the above equations, we can rewrite this ABP into a width-2 ABP whose underlying graph has no crossing edges, that is, a *planar* with-2 ABP. See Fig. 5 for an example with three Q-matrices.

4 VNP via products of affine linear forms

Valiant proved the following characterization of **VNP** [54] (see also [11, Thm. 21.26], [9, Thm. 2.13] and [36, Thm. 2]).

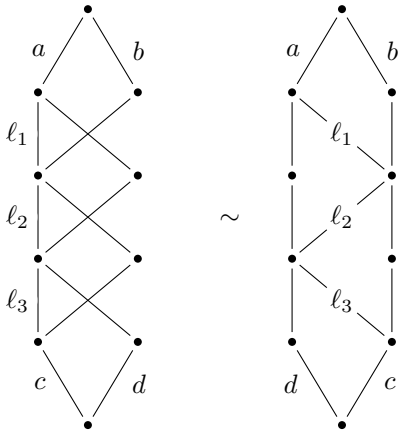


Figure 5 Making an ABP consisting of three primitive Q-matrices planar.

▶ **Theorem 4.1** (Valiant [54]). $\mathbf{VNP}_{\mathrm{e}} = \mathbf{VNP}$.

We strengthen Valiant's characterization of **VNP** from $\mathbf{VNP}_{\mathrm{e}}$ to $\mathbf{VNP}_1$.

▶ **Theorem 4.2.** $\mathbf{VNP}_1 = \mathbf{VNP}$ *when* $\mathrm{char}(\mathbb{F}) \neq 2$.

We give two proofs. The idea of the first proof is to show that the **VNP**-complete permanent family $\mathrm{per}_n := \sum_{\sigma\in S_n} \prod_{i\in[n]} x_{i,\sigma(i)}$ is in $\mathbf{VNP}_1$. The idea of the second proof is to simulate in $\mathbf{VNP}_1$ the primitives that are used in the proof of $\mathbf{VP}_{\mathrm{e}} = \mathbf{VP}_3$ by [6]. We present the second proof in Section 6. The advantage of the second proof is that we can restrict the ABP edge labels to affine linear forms that have at most 2 variables, see Theorem 6.2. Both proofs use the following lemma to write expressions of the form $1 + xy$ as a hypercube sum of a product of affine linear forms.

▶ **Lemma 4.3.** $\frac{1}{2}\sum_{b\in\{0,1\}}(x+1-2b)(y+1-2b) = 1+xy$ *when* $\mathrm{char}(\mathbb{F}) \neq 2$.

Proof. Expanding the left side gives the right side. ◀

Proof of Theorem 4.2. The permanent family (per_n) is well-known to be **VNP**-complete under p-projections, see for example [9, Thm. 2.10]. Therefore, to show that $\mathbf{VNP} \subseteq \mathbf{VNP}_1$, it suffices to show that $(\mathrm{per}_n) \in \mathbf{VNP}_1$. We begin by writing per_n as an inclusion-exclusion-type expression due to Ryser [45, Thm. 4.1],

$$\mathrm{per}_n = (-1)^n \sum_{S\subseteq[n]} (-1)^{|S|} \prod_{j\in[n]} \sum_{i\in S} x_{i,j}.$$

Encoding every subset $S \subseteq [n]$ by a bit string $b = (b[1], \ldots, b[n]) \in \{0,1\}^n$, we can rewrite the above as

$$\begin{aligned}\mathrm{per}_n &= (-1)^n \sum_{b\in\{0,1\}^n} \Big(\prod_{k\in[n]} (1-2b[k])\Big) \prod_{j\in[n]} \sum_{i\in[n]} b[i]\, x_{i,j} \\ &= (-1)^n \sum_{b\in\{0,1\}^n} \Big(\prod_{k\in[n]} (1-2b[k])\Big) \sum_{i_1,\ldots,i_n\in[n]} \prod_{j\in[n]} b[i_j]\, x_{i_j,j}\end{aligned}$$

For notational convenience we use square brackets not only to refer to sets ($[n] := \{1, \ldots, n\}$), but also to entries in a list ($b[k] := b_k$). We now introduce new Boolean variables $a[i,j]$, $1 \leq i \leq n-1$, $1 \leq j \leq n$, and fix the values $a[0,j] = 1$, $a[n,j] = 0$. (This gives an $(n+1) \times n$

matrix of variables and constants in which the first row consists of all 1s and the last row contains only 0s.) We claim that the above expression equals

$$\operatorname{per}_n = (-1)^n \sum_{b\in\{0,1\}^n} \Bigg(\prod_{k\in[n]} (1-2b[k]) \cdot \sum_a \prod_{i,j\in[n]} \big(1+(x_{i,j}-1)(a[i-1,j]-a[i,j])\big) \cdot \big(1+(b[i]-1)(a[i-1,j]-a[i,j])\big) \cdot \big(1+(a[i-1,j]-1)a[i,j]\big)\Bigg), \tag{3}$$

where the second sum is over all Boolean assignments of $a[i,j]$. The idea is to encode the indices $i_1,\dots,i_n$ in the boolean variables $a[i,j]$ in unary. For example, for $n=4$, if $i_1=4$, $i_2=3$, $i_3=1$, $i_4=4$, then the corresponding matrix a is

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

We prove the claim (3) in three steps. Fix j.

- If $a[i-1,j]=0$ and $a[i,j]=1$, then $1+(a[i-1,j]-1)a[i,j]=0$. Thus if in the sequence $a[0,j],\dots,a[n,j]$ a 0 is followed by a 1, then $\prod_{i\in[n]}(1+(a[i-1,j]-1)a[i,j]=0$. Conversely, if $(a[0,j],\dots,a[n,j])=(1,\dots,1,0,\dots,0)$, then $\prod_{i\in[n]}(1+(a[i-1,j]-1)a[i,j])=1$. The nontrivial assignments of $(a[0,j],\dots,a[n,j])$ are thus exactly of the form $(1,\dots,1,0,\dots,0)$ where the first 0 occurs at some index $1\le z\le n$ (since we have set $a[0,j]=1$ and $a[n,j]=0$). Fix such an assignment with first 0 occurring at index z.
- If $i=z$, then $1+(x_{i,j}-1)(a[i-1,j]-a[i,j])$ equals $x_{i,j}$. If $i\neq z$, it equals 1.
- If $i=z$, then $1+(b[i]-1)(a[i-1,j]-a[i,j])$ equals $b[i]$. If $i\neq z$, it equals 1.

This proves (3).

Next we apply Lemma 4.3, introducing fresh hypercube variables $c_1[i,j]$, $c_2[i,j]$, and $c_3[i,j]$, for $1\le i,j\le n$, to obtain

$$\begin{aligned}\operatorname{per}_n = (-1)^n (\tfrac{1}{2})^{3n^2} \sum_b \Big(\prod_{k\in[n]} (1-2\,b[k]) \Big) \cdot \sum_a \Bigg(\prod_{i,j\in[n]} & \sum_{c_1[i,j]} \Big[(x_{i,j}-2\,c_1[i,j]) \cdot (a[i-1,j]-a[i,j]+1-2\,c_1[i,j]) \Big] \\ \cdot & \sum_{c_2[i,j]} \Big[(b[i]-2\,c_2[i,j]) \cdot (a[i-1,j]-a[i,j]+1-2\,c_2[i,j]) \Big] \\ \cdot & \sum_{c_3[i,j]} \Big[(a[i-1,j]-2\,c_3[i,j]) \cdot (a[i,j]+1-2\,c_3[i,j]) \Big] \Bigg),\end{aligned}$$

where the sum goes over all Boolean assignments of $b[i]$, $a[i,j]$, $c_1[i,j]$, $c_2[i,j]$, $c_3[i,j]$, for all indices $1\le i,j\le n$, except for $a[n,j]\coloneqq 0$, and $a[0,j]\coloneqq 1$. After a rearrangement we obtain

the expression

$$\begin{aligned}\mathrm{per}_n = \sum_{\substack{a,b\\ c_1,c_2,c_3}} \Big((-1)^n(\tfrac{1}{2})^{3n^2}\Big(\prod_{k\in[n]}(1-2\,b[k])\Big)\cdot \prod_{i,j\in[n]} \\ (x_{i,j}-2\,c_1[i,j])\cdot(a[i{-}1,j]-a[i,j]+1-2\,c_1[i,j]) \\ \cdot(b[i]-2\,c_2[i,j])\cdot(a[i{-}1,j]-a[i,j]+1-2\,c_2[i,j]) \\ \cdot(a[i{-}1,j]-2\,c_3[i,j])\cdot(a[i,j]+1-2\,c_3[i,j])\Big),\end{aligned}$$

where the sum goes over all Boolean assignments of $a[i,j]$, $b[i]$, $c_1[i,j]$, $c_2[i,j]$, $c_3[i,j]$ for all indices $1 \le i,j \le n$, again except for $a[n,j] := 0$, and $a[0,j] := 1$. This shows that $(\mathrm{per}_n) \in \mathbf{VNP}_1$. ◀

In Section 9 we will prove that the statement of Theorem 4.2 does not hold over $\mathbb{F}_2$, that is, $\mathbf{VNP}_1 \subsetneq \mathbf{VNP}$ when $\mathbb{F} = \mathbb{F}_2$. We leave the situation over other fields of characteristic 2 as an open problem.

5 ABPs with restricted edge labels

So far the edge labels of our ABPs were allowed to be arbitrary affine linear forms. This section is about ABPs in which the edge labels are restricted to be simple affine linear forms ("weak ABPs"), or variables and constants ("weakest ABPs"). These edge label types were also studied in [2].

▶ **Definition 5.1.** A wst-ABP (weakest ABP) is an ABP with edges labeled by variables or constants. A w-ABP (weak ABP) is an ABP with edges labeled by simple affine linear forms $\alpha x_i + \beta$, $\alpha, \beta \in \mathbb{F}$. A g-ABP (general ABP) is an ABP with edges labeled by general affine linear forms $\sum_i \alpha_i x_i + \beta$, $\alpha_i, \beta \in \mathbb{F}$. For $*$ equal to wst, w or g, the class $\mathbf{VP}^*_k$ consists of all families of polynomials over polynomially many variables that are computed by polynomial-size width-k $*$-ABPs. In the rest of this paper the star will act as a variable from $\{\mathrm{wst}, \mathrm{w}, \mathrm{g}\}$. We write $\mathbf{VP}_k$ if we mean $\mathbf{VP}^{\mathrm{g}}_k$.

From the above definition it follows that $\mathbf{VP}^{\mathrm{wst}}_k \subseteq \mathbf{VP}^{\mathrm{w}}_k \subseteq \mathbf{VP}^{\mathrm{g}}_k$.

▶ Remark. One checks that the construction in the proof of Theorem 3.1 actually proves the inclusion $\mathbf{VP}_{\mathrm{e}} \subseteq \overline{\mathbf{VP}_2^{\mathrm{wst}}}^{\mathrm{poly}}$ when $\mathrm{char}(\mathbb{F}) \neq 2$. The inclusion $\mathbf{VP}_{\mathrm{e}} \subseteq \overline{\mathbf{VP}_2^{\mathrm{wst}}}^{\mathrm{poly}}$ implies the equalities $\overline{\mathbf{VP}_2^{\mathrm{wst}}} = \overline{\mathbf{VP}_{\mathrm{e}}}$ and $\overline{\mathbf{VP}_2^{\mathrm{wst}}}^{\mathrm{poly}} = \overline{\mathbf{VP}_{\mathrm{e}}}^{\mathrm{poly}}$.

In the following sections we will prove all inclusions and separations that are listed in Figure A.

5.1 Comparing different types of edge labels in width-2 ABPs

The aim of this subsection is to prove the following separation.

▶ **Theorem 5.2.** $\mathbf{VP}^{\mathrm{w}}_2 \subsetneq \mathbf{VP}^{\mathrm{g}}_2$.

In fact, we will show the following stronger statement.

▶ **Theorem 5.3.** *The polynomial*

$$\begin{aligned}p(\mathbf{x}) = (x_{11}+x_{12}+\cdots+x_{17})(x_{21}+x_{22}+\cdots+x_{27}) \\ +(x_{31}+x_{32}+\cdots+x_{37})(x_{41}+x_{42}+\cdots+x_{47})\end{aligned}$$

is computable by a width-2 g*-ABP, but not computable by any width-2* w*-ABP.*

We leave it as an open problem whether the inclusion $\mathbf{VP}_2^{\mathrm{wst}} \subseteq \mathbf{VP}_2^{\mathrm{w}}$ is strict.

To prove Theorem 5.3 we will review and reuse the arguments used by Allender and Wang [2] to show that the polynomial $x_1x_2+\cdots+x_{15}x_{16}$ cannot be computed by any width-2 g-ABP.

For the proof of Theorem 5.3 we may without loss of generality assume that the base field $\mathbb{F}$ is algebraically closed, because for any field $\mathbb{F}$, if p is not computable over the algebraic closure of $\mathbb{F}$, then it is not computable over $\mathbb{F}$ itself. Let $\mathbb{H}$ be the affine linear forms that are single variables x_i or constants $\mathbb{F}$. Let $\mathbb{S}$ be the set of simple affine linear forms. Let $\mathbb{L}$ be the set of general affine linear forms. Let $\mathbb{H}^{2\times 2}$, $\mathbb{S}^{2\times 2}$, $\mathbb{L}^{2\times 2}$ be the sets of 2×2 matrices with entries in $\mathbb{H}$, $\mathbb{S}$, $\mathbb{L}$ respectively. In this subsection, all ABPs have width 2, and by a wst-, w- or g-ABP Γ we will mean a sequence $\Gamma_k,\dots,\Gamma_1$ with $\Gamma_k \in \mathbb{F}^{1\times 2}$, $\Gamma_{k-1},\dots,\Gamma_2 \in X^{2\times 2}$, and $\Gamma_1 \in \mathbb{F}^{2\times 1}$ with X equal to $\mathbb{H}$, $\mathbb{S}$ or $\mathbb{L}$ respectively. We call $\Gamma_{k-1},\dots,\Gamma_2$ the *inner matrices of* Γ.

▶ **Definition 5.4.** A matrix $A \in \mathbb{L}^{2\times 2}$ is called *inherently nondegenerate* (indg) when $\det(A) \in \mathbb{F}\setminus\{0\}$.

Allender and Wang prove the following necessary condition for a polynomial to be computable by a wst-, w- or g-ABP whose inner matrices are indg. Let $\mathrm{H}(p)$ denote the highest-degree homogeneous part of a polynomial p.

▶ **Theorem 5.5** ([2, Thm. 3.9 and Lem. 4.7])**.** *Let p be a polynomial and Γ a* wst-, w- *or* g*-ABP computing p, whose inner matrices are indg. Then $\mathrm{H}(p)$ is a product of affine linear forms.*

Our next goal is to give a necessary condition for a polynomial p to be computable by a w-ABP. We begin with a simple lemma, which can essentially be found in [2].

▶ **Lemma 5.6** ([2])**.** *Let p be a polynomial. If p is computed by a* w*-ABP that has an inner matrix containing 4 variables, then there is an assignment π of 4 variables with $\pi(p)=0$.*

Proof. Let M be such a matrix. Since the ABP is of type w, M is of the form

$$M = \begin{pmatrix} \alpha_{11}x_{11}+\beta_{11} & \alpha_{12}x_{12}+\beta_{12} \\ \alpha_{21}x_{21}+\beta_{21} & \alpha_{22}x_{22}+\beta_{22} \end{pmatrix}$$

for some constants $\alpha_{ij} \in \mathbb{F}\setminus\{0\}$, $\beta_{ij} \in \mathbb{F}$. Applying the four assignments $x_{ij} \mapsto -\beta_{ij}/\alpha_{ij}$ makes M zero and thus p zero. ◀

We need two more ideas before we will state and prove the necessary condition we are after. (1) Let $A \in \mathbb{L}^{2\times 2}$ be nonzero and not-indg (that is, $\det(A)$ is either 0 or a nonconstant polynomial). Then there is an assignment π of the variables such that $\pi(A)$ has only constant entries and has rank 1. (2) Let p be a polynomial computed by an ABP Γ, that is, $p = \Gamma_k\cdots\Gamma_1$. Suppose that Γ contains a matrix Γ_i with only constant entries and with rank 1. Then there is a 2×1 matrix $\Gamma_{i,2}$ and a 1×2 matrix $\Gamma_{i,1}$ such that $\Gamma_i = \Gamma_{i,2}\Gamma_{i,1}$. Then p is a product

$$p = p_2p_1$$

of polynomials p_1, p_2, each computable by an ABP, namely

$$p_2 = \Gamma_k\cdots\Gamma_{i+1}\Gamma_{i,2}$$
$$p_1 = \Gamma_{i,1}\Gamma_{i-1}\cdots\Gamma_1.$$

We say that Γ_i *factors* p *into* p_2p_1. Recall that $\mathrm{H}(p)$ denotes the highest-degree homogeneous part of a polynomial p. The following is implicit in [2].

▶ **Theorem 5.7** ([2]). *Let p be a polynomial computed by a* w*-ABP Γ. Then there is an assignment π of at most 6 variables such that one of the following is true:*

1. *$\pi(p)$ is affine linear (including constant), or*
2. *$\mathrm{H}(\pi(p))$ is a product of two polynomial of positive degree.*

Proof. Let $(\Gamma_k, \ldots, \Gamma_1)$ be the matrices of Γ, so that $p = \Gamma_k \cdots \Gamma_1$. If there is a Γ_i containing 4 variables, then there is an assignment π of these 4 variables with $\pi(p) = 0$ (Lemma 5.6), so we are in case 1. Otherwise, all Γ_i have at most 3 variables. If the inner Γ_i are all indg, then $\mathrm{H}(p)$ is a product of linear forms (Theorem 5.5), so we are in case 1 or 2. Otherwise, there is at least one not-indg inner matrix. Consider the nonempty subsequence $\mathcal{M} = (M_\ell, \ldots, M_1)$ of not-indg inner matrices. For each M_i there is an assignment π of at most 3 variables such that $\pi(M_i)$ has only constant entries and rank 1. We consider four possible situations.

1. *There is an $M \in \mathcal{M}$ and an assignment π of at most 3 variables such that $\pi(M)$ factors $\pi(p)$ into a product of two constants or a product of two polynomials with positive degree.* Then we are in case 1 or 2.
2. *There is an assignment π of at most 3 variables such that $\pi(M_1)$ factors $\pi(p)$ into p_2p_1 with p_2 a constant and p_1 not constant.* Then p_1 is computed by an ABP consisting of indg inner matrices (since M_1 is the right-most not-indg inner matrix) and hence $\mathrm{H}(p_1)$ is a product of linear forms (Theorem 5.5), so we are in case 1 or 2.
3. *There is an assignment π of at most 3 variables such that $\pi(M_\ell)$ factors $\pi(p)$ into p_2p_1 with p_2 not a constant and p_1 a constant.* Then p_2 is computed by an ABP consisting of indg inner matrices (since M_2 is the left-most not-indg inner matrix) and one proceeds as in the previous situation.
4. *Remaining situation.* In the remaining situation we do the following. Let M_i be the left-most matrix in $\mathcal{M}$ such that there is an assignment π of at most 3 variables such that $\pi(M_i)$ factors $\pi(p)$ into p_2p_1 with p_2 not a constant and p_1 constant. Then there is an assignment σ of at most 3 variables such that $\sigma(\pi(M_{i+1}))$ factors $p_2 = p_3p_4$ with p_3 constant and p_4 not constant. Then p_4 is computed by an ABP consisting of indg matrices, and so $\mathrm{H}(p_4)$ is a product of homogeneous linear forms. Therefore we are in case 1 or 2. ◀

▶ **Theorem 5.3** (repeated). *The polynomial*

$$p(\mathbf{x}) = (x_{11} + x_{12} + \cdots + x_{17})(x_{21} + x_{22} + \cdots + x_{27}) + (x_{31} + x_{32} + \cdots + x_{37})(x_{41} + x_{42} + \cdots + x_{47})$$

is computable by a width-2 g*-ABP, but not computable by any width-2* w*-ABP.*

Proof. Clearly $p(\mathbf{x})$ is computable by a width-2 g-ABP. Suppose $p(\mathbf{x})$ is computable by a width-2 w-ABP. Then by Theorem 5.7 there is an assignment π of at most 6 variables such that either $\pi(p)$ is affine linear or $\mathrm{H}(\pi(p))$ is a product of two polynomials of positive degree. The first option is impossible, because distinct variables do not cancel. So $\mathrm{H}(\pi(p))$ is a product of two polynomials of positive degree. With another assignment σ we can achieve that $\mathrm{H}(\sigma(\pi(p))$ is of the form $x_ix_j + x_kx_\ell$ for some distinct variables x_i, x_j, x_k, x_ℓ. This is not a product of two polynomials of positive degree, so $\mathrm{H}(\pi(p))$ is not either. ◀

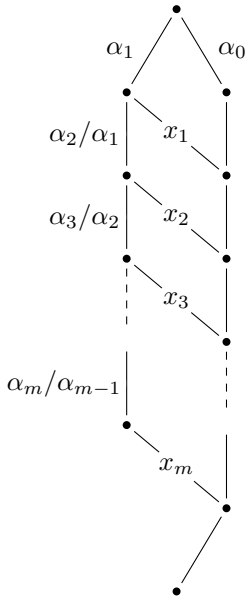


Figure 6 Width-2 wst-ABP computing $\ell(\mathbf{x}) = \alpha_0 + \alpha_1 x_1 + \alpha_2 x_2 + \cdots + \alpha_m x_m$.

5.2 Comparing different types of edge labels in width-1 ABPs

Clearly, $\mathbf{VP}_1^{\mathrm{wst}} \subseteq \mathbf{VP}_1^{\mathrm{w}} \subseteq \mathbf{VP}_1^{\mathrm{g}}$ and $\mathbf{VP}_1^{*} \subseteq \mathbf{VP}_2^{*}$, but this does not give a complete description of all inclusions among these classes. The following two propositions realize a complete description among $\mathbf{VP}_1^{*}$ and $\mathbf{VP}_2^{\mathrm{wst}}$.

▶ **Proposition 5.8.** $\mathbf{VP}_1^{\mathrm{g}} \subseteq \mathbf{VP}_2^{\mathrm{wst}}$.

Proof. Let $(p_n) \in \mathbf{VP}_1^{\mathrm{g}}$. Then each p_n is a product of $\mathrm{poly}(n)$ affine linear forms in $\mathrm{poly}(n)$ variables. Let $\ell(\mathbf{x}) = \alpha_0 + \alpha_1 x_1 + \alpha_2 x_2 + \cdots + \alpha_m x_m$ be such an affine linear form with $\alpha_0 \in \mathbb{F}$ and $\alpha_1, \ldots, \alpha_m \in \mathbb{F} \setminus \{0\}$. We can compute $\ell(\mathbf{x})$ with the width-2 wst-ABP in Fig. 6. A product of affine linear forms can be computed by the width-2 wst-ABP that is the concatenation of the width-2 wst-ABPs computing the affine linear forms. For p_n the resulting ABP has $\mathrm{poly}(n)$ size. Thus, $(p_n) \in \mathbf{VP}_2^{\mathrm{wst}}$. ◀

▶ **Proposition 5.9.** $\mathbf{VP}_1^{\mathrm{wst}} \subsetneq \mathbf{VP}_1^{\mathrm{w}} \subsetneq \mathbf{VP}_1^{\mathrm{g}} \subsetneq \mathbf{VP}_2^{\mathrm{wst}}$.

Proof. If $(p_n) \in \mathbf{VP}_1^{\mathrm{wst}}$, then p_n is a monomial. However, $(\alpha_0 + \alpha_1 x_1) \in \mathbf{VP}_1^{\mathrm{w}}$ and $\alpha_0 + \alpha_1 x_1$ is not a monomial, so $\mathbf{VP}_1^{\mathrm{wst}} \subsetneq \mathbf{VP}_1^{\mathrm{w}}$. If $(p_n) \in \mathbf{VP}_1^{\mathrm{w}}$ and p_n is homogeneous, then p_n is a monomial. However, $\left((x_1 + x_2)^2\right) \in \mathbf{VP}_1^{\mathrm{g}}$ and $(x_1 + x_2)^2$ is not a monomial, so $\mathbf{VP}_1^{\mathrm{w}} \subsetneq \mathbf{VP}_1^{\mathrm{g}}$. The last inclusion is Proposition 5.8. To see the strictness, if $(p_n) \in \mathbf{VP}_1^{\mathrm{g}}$, then the highest-degree homogeneous part $\mathrm{H}(p_n)$ of p_n is a product of homogeneous linear forms. However, $(x_1 x_2 + x_3 x_4) \in \mathbf{VP}_2^{\mathrm{wst}}$ and $x_1 x_2 + x_3 x_4$ is not a product of homogeneous linear forms, so $\mathbf{VP}_1^{\mathrm{g}} \subsetneq \mathbf{VP}_2^{\mathrm{wst}}$. ◀

5.3 Approximation in width-1 ABPs

The following proposition says that each of $\mathbf{VP}_1^{\mathrm{wst}}$, $\mathbf{VP}_1^{\mathrm{w}}$ and $\mathbf{VP}_1^{\mathrm{g}}$ is closed under approximation.

▶ **Proposition 5.10.** $\mathbf{VP}_1^{*} = \overline{\mathbf{VP}_1^{*}}$.

Proof. Trivially, $\mathbf{VP}_1^* \subseteq \overline{\mathbf{VP}_1^*}$. To prove the opposite inclusion, let $(f_n) \in \overline{\mathbf{VP}_1^*}$. There are polynomials $g_n(\varepsilon, \mathbf{x}) \in \mathbb{F}[\varepsilon, \mathbf{x}]$ such that $f_n + \varepsilon g_n(\varepsilon, \mathbf{x})$ can be written as a product of $\mathrm{poly}(n)$ affine linear forms in $\mathbb{F}(\varepsilon)[\mathbf{x}]$ in $\mathrm{poly}(n)$ variables (these affine linear forms have either wst-, w- or g-type). That is, (forgetting the subscript n for the moment) $f(\mathbf{x}) + \varepsilon g(\varepsilon, \mathbf{x})$ can be written as

$$f(\mathbf{x}) + \varepsilon g(\varepsilon, \mathbf{x}) = \prod_{i=1}^{m} \ell_i(\varepsilon, \mathbf{x})$$

with

$$\ell_i(\varepsilon, \mathbf{x}) = \sum_{j=d_i}^{e_i} \varepsilon^j k_{i,j}(\mathbf{x})$$

for some affine linear forms $k_{i,j} \in \mathbb{F}[\mathbf{x}]$, such that $k_{i,d_i}(\mathbf{x}) \neq 0$, and $d_i \leq e_i \in \mathbb{Z}$. By shifting ε-factors from $\ell_1, \ldots, \ell_{m-1}$ to ℓ_m we can assume that $d_i = 0$ for $i < m$. We claim that $d_m \geq 0$. If $d_m < 0$, then expanding $\prod_i \ell_i(\mathbf{x})$ as a Laurent series in ε gives a term with a negative power of ε. This contradicts $f(\mathbf{x}) + \varepsilon g(\mathbf{x})$ having only nonnegative powers of ε. Therefore, the $\ell_i(\mathbf{x})$ do not contain any negative powers of ε and we can safely substitute $\varepsilon \mapsto 0$ in each linear form ℓ_i to obtain f as a product of affine linear forms in $\mathbb{F}[\mathbf{x}]$ (either of wst-, w- or g-type). Remembering our subscript n again, we have thus proven $(f_n) \in \mathbf{VP}_1^*$. ◀

5.4 Nondeterminism in width-1 ABPs

In the following proposition we compare $\mathbf{VP}_1^*$ to $\mathbf{VNP}_1^*$ for all three versions $* \in \{\mathrm{wst}, \mathrm{w}, \mathrm{g}\}$.

▶ **Proposition 5.11.**

- $\mathbf{VP}_1^* = \mathbf{VNP}_1^*$ *for* $*$ *equal to* wst *or* w.
- $\mathbf{VP}_1^{\mathrm{g}} \subsetneq \mathbf{VNP}_1^{\mathrm{g}}$ *when* $\mathrm{char}(\mathbb{F}) \neq 2$.

Proof. Trivially, $\mathbf{VP}_1^* \subseteq \mathbf{VNP}_1^*$. Let $(p_n) \in \mathbf{VNP}_1^{\mathrm{wst}}$. Then p_n can be written as a hypercube-sum over a monomial,

$$p(\mathbf{x}) = \sum_{\mathbf{b} \in \{0,1\}^{\mathrm{poly}(n)}} m(\mathbf{b}, \mathbf{x})$$

with m a monomial (subscripts n are implied). For any $\mathbf{b}$-variable that does not occur in m, we remove that $\mathbf{b}$-variable form the summation and at the same time multiply the expression by 2, to again have an expression for $p(\mathbf{x})$. Assuming all $\mathbf{b}$-variables occur in m, only for $\mathbf{b} = (1, 1, \ldots, 1)$ can $m(\mathbf{b}, \mathbf{x})$ be nonzero. So $p(\mathbf{x}) = m((1, \ldots, 1), \mathbf{x})$. Remembering the subscript n, we proved $(p_n) \in \mathbf{VP}_1^{\mathrm{wst}}$.

Let $(p_n) \in \mathbf{VNP}_1^{\mathrm{w}}$. Then, (forgetting the subscript n)

$$p(\mathbf{x}) = \sum_{\mathbf{b} \in \{0,1\}^{\mathrm{poly}(n)}} \prod_i \ell_i(\mathbf{b}) \prod_j k_j(\mathbf{x})$$

for some simple affine linear forms ℓ_i in the variables $\mathbf{b}$ and some simple affine linear forms k_j in the variables $\mathbf{x}$. The product $\prod_j k_j(\mathbf{x})$ is independent of $\mathbf{b}$, while $\sum_{\mathbf{b}} \prod_i \ell_i(\mathbf{b})$ is a constant. We can thus write $p(\mathbf{x})$ as a constant times $\prod_j k_j(\mathbf{x})$. Therefore (remembering n), $p_n(\mathbf{x}) \in \mathbf{VP}_1^{\mathrm{w}}$. This proves the first line of the proposition.

To prove the second line, recall that if $(p_n) \in \mathbf{VP}_1^{\mathrm{g}}$, then p_n is a product of affine linear forms. However, let $p_n(x_1, x_2) = \sum_{b \in \{0,1\}} (x_1 + b)(x_2 + b) = 2x_1x_2 + x_1 + x_2 + 1$. Then

$(p_n) \in \mathbf{VNP}_1^{\mathrm{g}}$, but $p_n(x_1, x_2)$ is a not a product of affine linear forms, as we will now verify. Suppose $2x_1x_2 + x_1 + x_2 + 1 = (\alpha_0 + \alpha_1 x_1 + \alpha_2 x_2)(\beta_0 + \beta_1 x_1 + \beta_2 x_3)$. Then $\alpha_1\beta_1 = 0$ and $\alpha_2\beta_2 = 0$. Since $\alpha_1\beta_1 = 0$, we may assume without loss of generality that $\alpha_1 = 0$. Since not both α_1 and α_2 can be 0 (otherwise $(\alpha_0 + \alpha_1 x_1 + \alpha_2 x_2)(\beta_0 + \beta_1 x_1 + \beta_2 x_2)$ has degree 1) and since $\alpha_2\beta_2 = 0$, we have $\beta_2 = 0$. Hence, $2x_1x_2 + x_1 + x_2 + 1 = (\alpha_0 + \alpha_2 x_2)(\beta_0 + \beta_1 x_1)$. Then $\alpha_0\beta_0 = 1$, $\alpha_0\beta_1 = 1$, $\alpha_2\beta_0 = 1$, and $\alpha_2\beta_1 = 2$. The first two of these equations imply $\beta_0 = \beta_1$, which contradicts the last two of these equations. So $\mathbf{VP}_1^{\mathrm{g}} \subsetneq \mathbf{VNP}_1^{\mathrm{g}}$. ◀

▶ Remark 5.12. It follows directly from Proposition 5.11 and Proposition 5.9 that we have strict inclusions $\mathbf{VNP}_1^{\mathrm{wst}} \subsetneq \mathbf{VNP}_1^{\mathrm{w}} \subsetneq \mathbf{VNP}_1^{\mathrm{g}}$, when $\mathrm{char}(\mathbb{F}) \neq 2$.

6 Alternative proof of $\mathbf{VNP_1} = \mathbf{VNP}$ via $\mathbf{VP_3}$

Recall that in Section 4 we proved that

$$\mathbf{VNP}_1^{\mathrm{g}} = \mathbf{VNP} \tag{4}$$

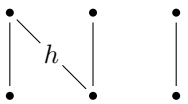
using the completeness of the permanent (Theorem 4.2). We will present an alternative proof of (4) inspired by the proof of the following theorem by Ben-Or and Cleve. The alternative proof of (4) has the benefit that it can be extended to show a slightly stronger result, see Theorem 6.2.

▶ **Theorem 6.1** (Ben-Or and Cleve, [6]). *For $k \geq 3$, $\mathbf{VP}_k^* = \mathbf{VP}_{\mathrm{e}}$.*

Proof. Proposition 7.1 says that $\mathbf{VP}_k^* \subseteq \mathbf{VP}_{\mathrm{e}}$. We will prove that $\mathbf{VP}_{\mathrm{e}} \subseteq \mathbf{VP}_3^{\mathrm{wst}}$, from which it follows that $\mathbf{VP}_{\mathrm{e}} \subseteq \mathbf{VP}_k^*$ and thus $\mathbf{VP}_k^* = \mathbf{VP}_{\mathrm{e}}$. For a polynomial h, define the matrix

$$M(h) := \begin{pmatrix} 1 & 0 & 0 \\ h & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

which, as part of an ABP, looks like



We call the following matrices *primitive*:

- $M(h)$ with h any variable or any constant in $\mathbb{F}$
- every 3×3 permutation matrix M_π with $\pi \in S_3$ any permutation
- every diagonal matrix $M_{a,b,c} := \mathrm{diag}(a, b, c)$ with a, b, c any constants in $\mathbb{F}$

The entries of the primitives are variables or constants in $\mathbb{F}$, making them suitable to use in the construction of a width-3 wst-ABP (Definition 5.1).

Let $(f_n) \in \mathbf{VP}_{\mathrm{e}}$. Then f_n can be computed by a formula of size $s(n) \in \mathrm{poly}(n)$. By Brent's depth-reduction theorem for formulas ([8]) f_n can then also be computed by a formula of size $\mathrm{poly}(n)$ and depth $d(n) \in \mathcal{O}(\log n)$.

We will construct a sequence of primitives $A_1, \ldots, A_{m(n)}$ such that

$$A_1 \cdots A_{m(n)} = \begin{pmatrix} 1 & 0 & 0 \\ f_n & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

with $m(n) \in \mathcal{O}(4^{d(n)}) = \text{poly}(n)$. Then

$$f_n(\mathbf{x}) = (\,1\;1\;1\,) M_{-1,1,0} A_1 \cdots A_m \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix},$$

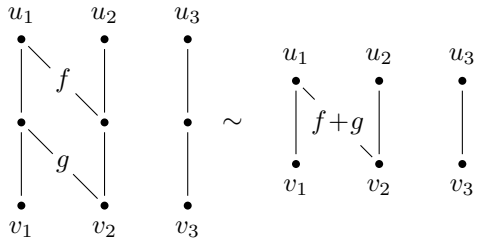
so $f_n(\mathbf{x})$ can be computed by a width-3 wst-ABP of size poly(n), proving the theorem.

To explain the construction, let h be a polynomial and consider a formula computing h of depth d. The goal is to construct (recursively on the formula structure) primitives $A_1, \ldots, A_m$ such that

$$A_1 \cdots A_m = \begin{pmatrix} 1 & 0 & 0 \\ h & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \text{with } m \in \mathcal{O}(4^d). \tag{5}$$

Suppose h is a variable or a constant. Then $M(h)$ is itself a primitive matrix.

Suppose $h = f + g$ is a sum of two polynomials f, g and suppose $M(f)$ and $M(g)$ can be written as a product of primitives. Then $M(f+g)$ equals a product of primitives, because $M(f+g) = M(f)M(g)$. This can easily be verified directly, or by noting that in the corresponding partial ABPs the top-bottom paths (u_i-v_j paths) have the same value:

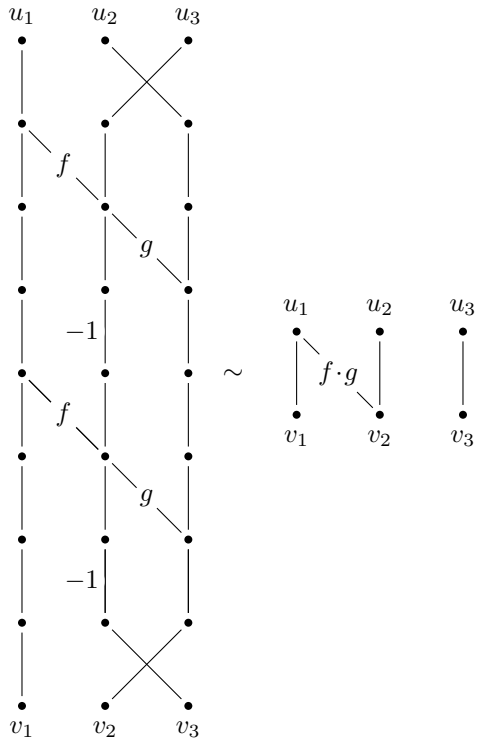


Suppose $h = fg$ is a product of two polynomials f, g and suppose $M(f)$ and $M(g)$ can be written as a product of primitives. Then $M(fg)$ equals a product of primitives, because

$$M(f \cdot g) = M_{(23)} \big(M_{1,-1,1} M_{(123)} M(g) M_{(132)} M(f) \big)^2 M_{(23)}$$

(here $(23) \in S_3$ denotes the transposition $1 \mapsto 1, 2 \mapsto 3, 3 \mapsto 2$ and $(123) \in S_3$ denotes the cyclic shift $1 \mapsto 2, 2 \mapsto 3, 3 \mapsto 1$) as can be verified either directly or by checking that in the

corresponding partial ABPs the top-bottom paths (u_i-v_j paths) have the same value:



This completes the construction.

The length m of the construction is $m(h) = 1$ for h a variable or constant and recursively $m(f + g) = m(f) + m(g)$, $m(f \cdot g) = 2(m(f) + m(g)) + \mathcal{O}(1)$, so $m \in \mathcal{O}(4^d)$ where d is the formula depth of h. The construction thus satisfies (5), proving the theorem. ◀

We will now give an alternative proof of Theorem 4.2.

▶ **Theorem 4.2** (repeated). $\mathbf{VNP}_1 = \mathbf{VNP}$ *when* $\mathrm{char}(\mathbb{F}) \neq 2$.

Proof. Clearly, $\mathbf{VNP}_1^{\mathrm{g}} \subseteq \mathbf{VNP}$ by Proposition 7.1 and taking the nondeterminism closure N. We will prove that $\mathbf{VNP} \subseteq \mathbf{VNP}_1^{\mathrm{g}}$.

Recall that in the proof of $\mathbf{VP}_{\mathrm{e}} \subseteq \mathbf{VP}_3^{\mathrm{wst}}$ (Theorem 6.1), we defined for any polynomial h the matrix

$$M(h) := \begin{pmatrix} 1 & 0 & 0 \\ h & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

and we called the following matrices *primitive*:

- $M(h)$ with h any variable or any constant in $\mathbb{F}$
- every 3×3 permutation matrix M_π with $\pi \in S_3$ any permutation
- every diagonal matrix $M_{a,b,c} := \mathrm{diag}(a, b, c)$ with a, b, c any constants.

In the proof of $\mathbf{VP}_{\mathrm{e}} \subseteq \mathbf{VP}_3^{\mathrm{wst}}$ we constructed, for any family $(f_n) \in \mathbf{VP}_{\mathrm{e}}$ a sequence of primitives $A_{n,1}, \ldots, A_{n,t(n)}$ with $t(n) \in \mathrm{poly}(n)$ such that

$$f_n(\mathbf{x}) = (1\ 1\ 1) M_{-1,1,0} A_1 \cdots A_m \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}.$$

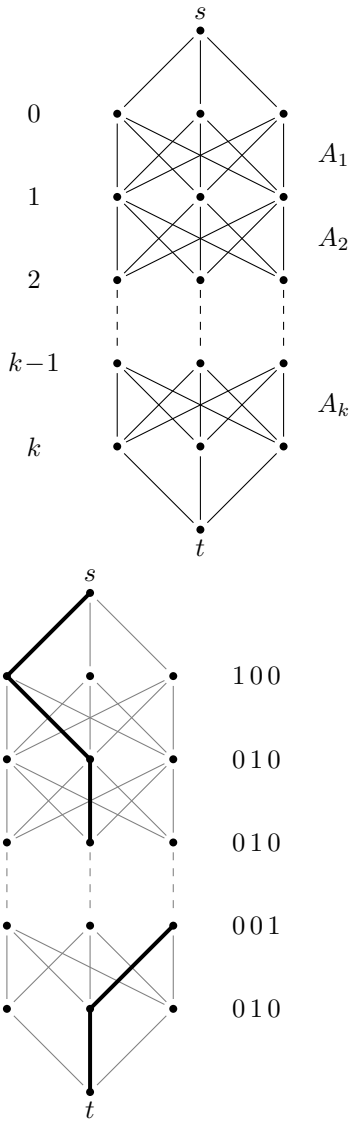


Figure 7 Illustration of layer labelling and path labelling in the proof of Theorem 4.2.

We will construct a hypercube sum over a width-1 g-ABP that evaluates the right-hand side, to show that $\mathbf{VP}_{\mathrm{e}} \subseteq \mathbf{VNP}_1^{\mathrm{g}}$. This implies $\mathbf{VNP}_{\mathrm{e}} \subseteq \mathbf{VNP}_1^{\mathrm{g}}$. Then by Valiant's Theorem 4.1, $\mathbf{VNP} \subseteq \mathbf{VNP}_1^{\mathrm{g}}$.

Let $f(\mathbf{x})$ be a polynomial and let $A_1, \ldots, A_k$ be primitives such that $f(\mathbf{x})$ is computed as

$$f(\mathbf{x}) = (1\ 1\ 1)A_k \cdots A_1 \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}.$$

View this expression as a width-3 ABP G, with vertex layers labeled as shown in the left diagram of Fig. 7.

Assume for simplicity that all edges between layers are present, possibly with label 0. The sum of the values of every s-t path in G equals $f(\mathbf{x})$,

$$f(\mathbf{x}) = \sum_{j \in [3]^k} A_k[j_k, j_{k-1}] \cdots A_1[j_2, j_1]. \tag{6}$$

We now introduce some hypercube variables. To every vertex, except s and t, we associate a bit; the bits in the ith layer we call $b_1[i]$, $b_2[i]$, $b_3[i]$. To an s-t path in G we associate an assignment of the $b_j[i]$ by setting the bits of vertices visited by the path to 1 and the others to 0. For example, in the right diagram in Fig. 7 we show an s-t path with the corresponding assignment of the bits $b_1[i]$, $b_2[i]$, and $b_3[i]$. The assignments of $b_j[i]$ corresponding to s-t paths are the ones such that for every $i \in [k]$ exactly one of $b_1[i]$, $b_2[i]$, $b_3[i]$ equals 1. Let

$$V(b_1, b_2, b_3) := \prod_{i\in[k]} (b_1[i] + b_2[i] + b_3[i]) \prod_{\substack{s,t\in[3]:\\ s\neq t}} \big(1 - b_s[i]b_t[i]\big). \tag{7}$$

The assignments of $b_j[i]$ corresponding to s-t paths are thus the ones such that $V(b_1, b_2, b_3) = 1$. Otherwise, $V(b_1, b_2, b_3) = 0$.

We will now write $f(\mathbf{x})$ as a hypercube sum by replacing each $A_i[j_i, j_{i-1}]$ in (6) by a product of affine linear forms $S_i(A_i)$ with variables $\mathbf{b}$ and $\mathbf{x}$ as follows

$$\sum_{\mathbf{b}} V(b_1, b_2, b_3) S_k(A_k) \cdots S_1(A_1).$$

Define $\mathrm{Eq}(\alpha, \beta) : \{0,1\}^2 \to \{0,1\}$ by $(1 - \alpha - \beta)(1 - \alpha - \beta)$. This function is 1 if $\alpha = \beta$ and 0 otherwise.

- For any variable or constant x define

$$\begin{aligned} \mathrm{S}_i(M(x)) := {} & \big(1 + (x-1)(b_1[i] - b_1[i-1])\big) \\ & \cdot \big(1 - (1 - b_2[i])b_2[i-1]\big) \\ & \cdot \mathrm{Eq}\big(b_3[i-1], b_3[i]\big). \end{aligned}$$

- For any permutation $\pi \in S_3$ define

$$\begin{aligned} \mathrm{S}_i(M_\pi) := {} & \mathrm{Eq}\big(b_1[i-1], b_{\pi(1)}[i]\big) \\ & \cdot \mathrm{Eq}\big(b_2[i-1], b_{\pi(2)}[i]\big) \\ & \cdot \mathrm{Eq}\big(b_3[i-1], b_{\pi(3)}[i]\big). \end{aligned}$$

- For any constants $a, b, c \in \mathbb{F}$ define

$$\begin{aligned} \mathrm{S}_i(M_{a,b,c}) := {} & \big(a \cdot b_1[i-1] + b \cdot b_2[i-1] + c \cdot b_3[i-1]\big) \\ & \cdot \mathrm{Eq}\big(b_1[i-1], b_1[i]\big) \\ & \cdot \mathrm{Eq}\big(b_2[i-1], b_2[i]\big) \\ & \cdot \mathrm{Eq}\big(b_3[i-1], b_3[i]\big). \end{aligned}$$

One verifies that with these definitions indeed

$$f(\mathbf{x}) = \sum_{\mathbf{b}} V(b_1, b_2, b_3) S_k(A_k) \cdots S_1(A_1).$$

Some of the factors in the $S_i(A_i)$ are not affine linear. As a final step we apply the equation $1 + xy = \frac{1}{2}\sum_{c\in\{0,1\}}(x + 1 - 2c)(y + 1 - 2c)$ (Lemma 4.3) to write these factors as products of affine linear forms, introducing new hypercube variables. ◀

Combining Theorem 4.2 and Remark 5.12 gives the separation $\mathbf{VNP}_1^{\mathrm{w}} \subsetneq \mathbf{VNP}_1^{\mathrm{g}} = \mathbf{VNP}$. We can prove a slightly stronger separation by adjusting the construction in the above proof

of Theorem 4.2. Namely, let $\mathbb{S}^+ := \{\alpha x_i + \beta x_j + \gamma \mid \alpha, \beta, \gamma \in \mathbb{F}\}$ be the set of affine linear forms in at most two variables and let $\mathbf{VP}_1^{\mathrm{w}+}$ be the class of families that can be computed by width-1 ABPs over $\mathbb{S}^+$ of polynomial size. Define $\mathbf{VNP}_1^{\mathrm{w}+}$ accordingly (Definition 2.2). Then we can adjust the construction in the above proof of Theorem 4.2 to show the following.

▶ **Theorem 6.2.** $\mathbf{VNP}_1^{\mathrm{w}} \subsetneq \mathbf{VNP}_1^{\mathrm{w}+} = \mathbf{VNP}$ *when* $\mathrm{char}(\mathbb{F}) \neq 2$.

Proof. We only need to show $\mathbf{VNP}_1^{\mathrm{w}+} = \mathbf{VNP}$, as $\mathbf{VNP}_1^{\mathrm{w}} \subsetneq \mathbf{VNP}$ was shown in Remark 5.12. The adjustments we have to make to the construction in the proof of Theorem 4.2 are as follows. Most of the resulting polynomial of the construction is already of the correct form where each linear forms contains at most two variables, since the expression $\mathrm{Eq}(x, y) = (1 - x - y)^2$ and the expression $1 + xy = \frac{1}{2}\sum_{c\in\{0,1\}}(x + 1 - 2c)(y + 1 - 2c)$ are of this form. Three expressions occur that are not of the correct form:

1. $b_1[i] + b_2[i] + b_3[i]$ in $V(b_1, b_2, b_3)$,
2. $a \cdot b_1[i-1] + b \cdot b_2[i-1] + c \cdot b_3[i-1]$ in $\mathrm{S}(M_{a,b,c})$, and
3. $1 + (x - 1)(b_1[i] - b_1[i-1])$ in $\mathrm{S}(M(x))$

Expression 1 and expression 2 we can write in the correct form using the identity

$$\tfrac{1}{2}\sum_{b\in\{0,1\}}(x + 1 - 2b)(y + 1 - 2b)(z + 1 - 2b) = x + y + z + xyz. \tag{8}$$

Indeed, expression 1 can be replaced by

$$\begin{aligned}&\tfrac{1}{2}\sum_{c\in\{0,1\}}(b_1[i] + 1 - 2c)(b_2[i] + 1 - 2c)(b_3[i] + 1 - 2c)\\ &\quad = b_1[i] + b_2[i] + b_3[i] + b_1[i]b_2[i]b_3[i],\end{aligned}$$

since the unwanted term $b_1[i]b_2[i]b_3[i]$ will always vanish in our construction (because in (7) we multiply with $1 - b_s[i]b_t[i]$ for every $s \neq t$). Similarly for expression 2.

For expression 3, we first replace the expression $1+(x-1)(b_1[i]-b_1[i-1])$ by the expression $\frac{1}{2}\sum_{c\in\{0,1\}}(x-1+1-2c)(b_1[i]-b_1[i-1]+1-2c)$. The second factor has too many variables. We replace it, using identity (8), by

$$\begin{aligned}&\tfrac{1}{2}\sum_{c'\in\{0,1\}}\big(b_1[i] + 1 - 2c'\big)\big(-b_1[i-1] + 1 + 1 - 2c'\big)\big(-2c + 1 - 2c'\big)\\ &\quad = b_1[i] - b_1[i-1] + 1 - 2c + b_1[i]\big(1 - b_1[i-1]\big)\big(-2c\big).\end{aligned}$$

The first four summands in the right-hand side are as we want. The last summand is only nonzero if $b_1[i] = 1$ and $b_1[i-1] = 0$. However, since $\mathrm{S}_i(M(x))$ contains a factor $1-(1-b_2[i])b_2[i-1]$ and a factor $\mathrm{Eq}(b_3[i-1], b_3[i])$, it can be checked that this last summand will always vanish.

In the new construction thus obtained each linear form is in $\mathbb{S}^+$. This completes the necessary adjustments to the construction. ◀

7 Constant-width ABPs have small formulas

The following well-known proposition says that the iterated product of constant-size matrices can be efficiently computed by a formula.

▶ **Proposition 7.1.** *Let* $k \geq 1$. *Then* $\mathbf{VP}_k \subseteq \mathbf{VP}_{\mathrm{e}}$.

Proof. Let $(f_n) \in \mathbf{VP}_k$, so f_n has $v(n) \in \mathrm{poly}(n)$ variables. There is a function $m(n) \in \mathrm{poly}(n)$ and there are $k \times k$ matrices $M_{n,1}, \ldots, M_{n,m(n)}$ with affine linear forms as entries, such that $\mathrm{tr}(M_{n,1} \cdots M_{n,m(n)}) = f_n$. We may assume that each affine linear form occurring in $M_{n,i}$ has $v(n)$ variables, since f_n has $v(n)$ variables. We will recursively construct a multi-output formula computing the product $M_{n,1} \cdots M_{n,m(n)}$. From this one can efficiently compute the trace. Let $\mathrm{size}(m, n)$ denote the size of the formula that we construct. Let $w(n) := 2v(n)$. A single matrix $M_{n,i}$ we can compute by a multi-output formula of size $k^2 w(n)$ (the $w(n)$ is needed to compute each affine linear form). So $\mathrm{size}(1, n) = k^2 w(n)$. Suppose that matrices A and B can be computed by a multi-output formulas F and G respectively, each of size s. Then the product AB can be computed by a multi-output formula of size $2ks + c(k)$ with $c(k) \in \mathcal{O}(k^3)$, as follows: take k copies of the formula F for A and take k copies of the formula G for B and appropriately add $c(k)$ $\times$- and $+$-gates in order to perform the matrix multiplication. (The reason that we take k copies of the formulas F and G is that in the matrix multiplication, each input entry is used k times, and in formulas we cannot use intermediate results more than once.) Therefore, the recurrence relation $\mathrm{size}(m, n) = 2k\,\mathrm{size}(m/2, n) + c(k)$ holds. Working out the recurrence relation gives $\mathrm{size}(m, n) = (2k)^{\log_2(m)} k^2 w(n) + \left[(2k)^{\log_2(m)} + (2k)^{\log_2(m)-1} + \cdots + 1\right] c(k)$. Since $v(n), m(n) \in \mathrm{poly}(n)$ and since k is constant in n, we have $\mathrm{size}(m(n), n) \in \mathrm{poly}(n)$. We can thus also compute the trace of $M_{n,1} \cdots M_{n,m(n)}$ with a $\mathrm{poly}(n)$-size formula. This shows that $(f_n) \in \mathbf{VP}_{\mathrm{e}}$. We thus have $\mathbf{VP}_k \subseteq \mathbf{VP}_{\mathrm{e}}$. ◀

8 Poly-approximation in width-2 ABPs

We give the interpolation argument that completes the proof of Corollary 3.9, which says that the poly-approximation closure of $\mathbf{VP}_2$ equals $\mathbf{VP}_{\mathrm{e}}$ when $\mathrm{char}(\mathbb{F}) \neq 2$ and $\mathbb{F}$ is infinite.

▶ **Proposition 8.1.** $\overline{\mathbf{VP}_{\mathrm{e}}}^{\mathrm{poly}} = \mathbf{VP}_{\mathrm{e}}$ *when* $\mathrm{char}(\mathbb{F}) \neq 2$ *and* $\mathbb{F}$ *is infinite.*

Proof. The inclusion $\mathbf{VP}_{\mathrm{e}} \subseteq \overline{\mathbf{VP}_{\mathrm{e}}}^{\mathrm{poly}}$ is clear. For the other direction, let $(f_n) \in \overline{\mathbf{VP}_{\mathrm{e}}}^{\mathrm{poly}}$. Then there are polynomials $f_{n;i}(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$, $e(n) \in \mathrm{poly}(n)$ such that

$$f_n(\mathbf{x}) + \varepsilon f_{n;1}(\mathbf{x}) + \varepsilon^2 f_{n;2}(\mathbf{x}) + \cdots + \varepsilon^{e(n)} f_{n;e(n)}(\mathbf{x})$$

is computed by a poly-size formula Γ over $\mathbb{F}(\varepsilon)$. Let $\alpha_0, \alpha_1, \ldots, \alpha_{e(n)}$ be distinct elements in $\mathbb{F}$ such that replacing ε by α_j in Γ is a valid substitution (these α_j exist since by assumption our field is infinite). View

$$g_n(\varepsilon) \coloneqq f_n(\mathbf{x}) + \varepsilon f_{n;1}(\mathbf{x}) + \varepsilon^2 f_{n;2}(\mathbf{x}) + \cdots + \varepsilon^{e(n)} f_{n;e(n)}(\mathbf{x})$$

as a polynomial in ε. The polynomial $g_n(\varepsilon)$ has degree at most $e(n)$ so we can write $g_n(\varepsilon)$ as follows (Lagrange interpolation on $e(n) + 1$ points)

$$g_n(\varepsilon) = \sum_{j=0}^{e(n)} g_n(\alpha_j) \prod_{\substack{0 \leq m \leq e(n): \\ m \neq j}} \frac{\varepsilon - \alpha_m}{\alpha_j - \alpha_m}. \tag{9}$$

Clearly, $f_n(\mathbf{x}) = g_n(0)$. From (9) we see directly how to write $g_n(0)$ as a linear combination of the values $g_n(\alpha_j)$, namely

$$g_n(0) = \sum_{j=0}^{e(n)} g_n(\alpha_j) \prod_{\substack{0 \leq m \leq e(n): \\ m \neq j}} \frac{-\alpha_m}{\alpha_j - \alpha_m},$$

that is,

$$g_n(0) = \sum_{j=0}^{e(n)} \beta_j \, g_n(\alpha_j) \quad \text{with} \quad \beta_j \coloneqq \prod_{\substack{0 \le m \le e(n): \\ m \neq j}} \frac{\alpha_m}{\alpha_m - \alpha_j}.$$

The value $g_n(\alpha_j)$ is computed by the formula Γ with ε replaced by α_j, which we denote by $\Gamma|_{\varepsilon=\alpha_j}$. Thus $f_n(\mathbf{x})$ is computed by the poly-size formula $\sum_{j=0}^{e(n)} \beta_j \, \Gamma|_{\varepsilon=\alpha_j}$. Therefore we have $(f_n) \in \mathbf{VP}_\mathrm{e}$. ◀

▶ Remark 8.2. Proposition 8.1 also holds with $\mathbf{VP}_\mathrm{e}$ replaced by $\mathbf{VP}_\mathrm{s}$ or $\mathbf{VP}$ by a similar proof.

9 $\mathbf{VNP}_1 \subsetneq \mathbf{VNP}$ when $\mathbb{F} = \mathbb{F}_2$

In our proofs of $\mathbf{VNP}_1 = \mathbf{VNP}$ (Section 4 and Section 6) the assumption $\mathrm{char}(\mathbb{F}) \neq 2$ played a crucial role. We can prove that over the finite field $\mathbb{F}_2$ the inclusion $\mathbf{VNP}_1 \subseteq \mathbf{VNP}$ is indeed strict.

▶ **Proposition 9.1.** $\mathbf{VNP}_1 \subsetneq \mathbf{VNP}$ *when* $\mathbb{F} = \mathbb{F}_2$.

Proof. Let $\mathbb{F} = \mathbb{F}_2$. Clearly $(1 + xy) \in \mathbf{VNP}$. However, we will prove that $1 + xy$ cannot be written as a hypercube sum of affine linear forms. In fact, we will prove something stronger, namely that the *function* $(x, y) \mapsto 1 + xy$ cannot be written as a hypercube sum of a product of affine linear forms.

Assume the contrary: the function $(x, y) \mapsto 1 + xy$ can be written as a hypercube sum of a product of affine linear forms. We can thus write

$$1 + xy = \textstyle\sum_{\mathbf{b}} L_{\mathbf{b}} \quad \text{with} \quad L_{\mathbf{b}} \coloneqq \prod_{i=1}^{\alpha}(x + A_i) \prod_{j=1}^{\beta}(y + B_j) \prod_{k=1}^{\gamma}(x + y + C_k) \tag{10}$$

for some affine linear forms $A_i(\mathbf{b})$, $B_j(\mathbf{b})$, $C_k(\mathbf{b})$ in the hypercube variables $\mathbf{b}$. On $\mathbb{F}_2$ the functions $x, x^2, x^3, \ldots$ coincide; the functions $y, y^2, y^3, \ldots$ coincide; and the functions $x + y, (x + y)^2, (x + y)^3, \ldots$ coincide, so

$$\begin{aligned}
\textstyle\prod_i (x + A_i) &= \textstyle\prod_i A_i + x\Big(\prod_i (1 + A_i) + \prod_i A_i\Big), \\
\textstyle\prod_j (y + B_j) &= \textstyle\prod_j B_j + y\Big(\prod_j (1 + B_j) + \prod_j B_j\Big), \\
\textstyle\prod_k (x + y + C_k) &= \textstyle\prod_k C_k + (x + y)\Big(\prod_k (1 + C_k) + \prod_k C_k\Big).
\end{aligned}$$

Multiplying the three expressions and simplifying powers of x and y gives

$$\begin{aligned}
L_{\mathbf{b}} = {} & \textstyle\prod_{i,j,k} A_i B_j C_k + x\Big(\prod_{i,j,k}(1 + A_i) B_j (1 + C_k) + \prod_{i,j,k} A_i B_j C_k\Big) \\
& + y\Big(\textstyle\prod_{i,j,k} A_i (1 + B_j)(1 + C_k) + \prod_{i,j,k} A_i B_j C_k\Big) \\
& + xy\Big(\textstyle\prod_{i,j,k} A_i (1 + B_j)(1 + C_k) + \prod_{i,j,k}(1 + A_i) B_j (1 + C_k) \\
& \qquad + \textstyle\prod_{i,j,k}(1 + A_i)(1 + B_j) C_k + \prod_{i,j,k} A_i B_j C_k\Big).
\end{aligned}$$

Plugging in the four possible assignments $(x, y) \in \mathbb{F}_2 \times \mathbb{F}_2$ into $1 + xy = \sum_{\mathbf{b}} L_{\mathbf{b}}$, we get the following system of equations

$$\sum_{\mathbf{b}} \prod_{i,j,k} A_i B_j C_k = 1, \tag{11}$$

$$\sum_{\mathbf{b}} \prod_{i,j,k} (1 + A_i) B_j (1 + C_k) = 1, \tag{12}$$

$$\sum_{\mathbf{b}} \prod_{i,j,k} A_i (1 + B_j)(1 + C_k) = 1, \tag{13}$$

$$\sum_{\mathbf{b}} \prod_{i,j,k} (1 + A_i)(1 + B_j) C_k = 0. \tag{14}$$

We will show that the above system of equations is inconsistent. Note that (11) asserts that an odd number of vectors $\mathbf{b}$ satisfy the system of equations

$$A_i = 1 \ \forall i$$
$$B_j = 1 \ \forall j$$
$$C_k = 1 \ \forall k.$$

Recall that we defined α, β, γ as the number of factors $x + A_i$, $y + B_j$, $x + y + C_k$ in (10), respectively. Let $m \coloneqq \alpha + \beta + \gamma$. Recall that we defined n as the number of hypercube variables b_ℓ. As we work over $\mathbb{F}_2$, any affine linear form in $\mathbf{b}$ can be written as $\alpha_0 + \sum_{\ell=1}^{n} \alpha_\ell b_\ell$ with $\alpha_i \in \{0, 1\}$. Write the ith linear form in $(A_1, \ldots, A_\alpha, B_1, \ldots, B_\beta, C_1, \ldots, C_\gamma)$ as $v_{0,i} + \sum_{\ell=1}^{n} b_\ell v_{\ell,i}$, and let $v_\ell = (v_{\ell,1}, \ldots, v_{\ell,m})$ for $0 \leq \ell \leq n$. We define the linear map $M : \mathbb{F}_2^n \to \mathbb{F}_2^m$ by $M(\mathbf{b}) = \sum_{\ell=1}^{n} b_\ell v_\ell$. We call a bit vector $\mathbf{b} \in \mathbb{F}_2^n$ a *solution* of (11) if $M(b) = v_0 + 1^\alpha 1^\beta 1^\gamma$, where $1^\alpha 1^\beta 1^\gamma$ is the all-ones vector. Observe that (11) says that there is an odd number of solutions of (11). Since the set of solutions of (11) forms an affine linear subspace of $(\mathbb{F}_2)^n$, its cardinality is a power of two. The only odd power of two is 1, so there is exactly one solution of (11). Let $b^{(1)}$ be this unique solution: $M(b^{(1)}) = v_0 + 1^\alpha 1^\beta 1^\gamma$. We do the same for (12) and (13) and find unique solutions $M(b^{(2)}) = v_0 + 0^\alpha 1^\beta 0^\gamma$ and $M(b^{(3)}) = v_0 + 1^\alpha 0^\beta 0^\gamma$. Equation (14) asserts that the number of solutions of (14) is even. One solution of (14) is given by $M(b^{(1)} + b^{(2)} + b^{(3)}) = 3v_0 + 1^\alpha 1^\beta 1^\gamma + 0^\alpha 1^\beta 0^\gamma + 1^\alpha 0^\beta 0^\gamma = v_0 + 0^\alpha 0^\beta 1^\gamma$. Let $b^{(4')}$ and $b^{(4'')}$ be two distinct solutions of (14) with $M(b^{(4')}) = M(b^{(4'')}) = v_0 + 0^\alpha 0^\beta 1^\gamma$. Then $M(b^{(2)} + b^{(3)} + b^{(4')}) = v_0 + 1^\alpha 1^\beta 1^\gamma = M(b^{(2)} + b^{(3)} + b^{(4'')})$, which contradicts the uniqueness of $b^{(1)}$. ◀

▶ Remark. Our proof of Proposition 9.1 does not generalize to all fields $\mathbb{F}$ of characteristic 2, because the polynomial $1 + xy$ is in fact computable by a hypercube sum of a product of affine linear forms when $\mathbb{F} = \mathbb{F}_4$ (and thus when $\mathbb{F} = \mathbb{F}_{2^{2k}}$, $k \in \mathbb{N}$). Indeed, $\mathbb{F}_4 \cong \mathbb{F}_2[Z]/(Z^2 + Z + 1)$, so the element $Z \in \mathbb{F}_4$ is a third root of unity ($Z^3 = 1$) and satisfies $Z^2 + Z + 1 = 0$. It can be checked that therefore $\sum_{b=0}^{1} (x + Z^2 y + Zb) \cdot (x + Zy + Z^2 b) \cdot (x + y + b)$ equals $1 + xy$.

References

1. Manindra Agrawal and V. Vinay. Arithmetic circuits: A chasm at depth four. *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 67–75, 2008. doi:10.1109/FOCS.2008.32.
2. Eric Allender and Fengming Wang. On the power of algebraic branching programs of width two. *Comput. Complexity*, 25(1):217–253, 2016. doi:10.1007/s00037-015-0114-7.
3. Jarod Alper, Tristram Bogart, and Mauricio Velasco. A lower bound for the determinantal complexity of a hypersurface. *Found. Comput. Math.*, pages 1–8, 2015. arXiv:1505.02205, doi:10.1007/s10208-015-9300-x.
4. Alexander E. Andreev. A method for obtaining more than quadratic effective lower estimates of complexity of π schemes. *Moscow Univ. Math. Bull.*, 42(1):63–66, 1987.

5 Daniel J. Bates and Luke Oeding. Toward a salmon conjecture. *Exp. Math.*, 20(3):358–370, 2011. doi:10.1080/10586458.2011.576539.

6 Michael Ben-Or and Richard Cleve. Computing algebraic formulas using a constant number of registers. *SIAM J. Comput.*, 21(1):54–58, 1992. doi:10.1137/0221006.

7 Dario Bini, Milvio Capovani, Francesco Romani, and Grazia Lotti. $O(n^{2.7799})$ complexity for $n \times n$ approximate matrix multiplication. *Inf. Process. Lett.*, 8(5):234–235, 1979. doi:10.1016/0020-0190(79)90113-3.

8 Richard P. Brent. The parallel evaluation of general arithmetic expressions. *J. ACM*, 21(2):201–206, April 1974. doi:10.1145/321812.321815.

9 Peter Bürgisser. *Completeness and reduction in algebraic complexity theory*, volume 7 of *Algorithms Comput. Math.* Springer-Verlag, Berlin, 2000. doi:10.1007/978-3-662-04179-6.

10 Peter Bürgisser. The complexity of factors of multivariate polynomials. *Found. Comput. Math.*, 4(4):369–396, 2004. doi:10.1007/s10208-002-0059-5.

11 Peter Bürgisser, Michael Clausen, and M. Amin Shokrollahi. *Algebraic complexity theory*, volume 315 of *Grundlehren Math. Wiss.* Springer-Verlag, Berlin, 1997. doi:10.1007/978-3-662-03338-8.

12 Peter Bürgisser and Christian Ikenmeyer. Geometric complexity theory and tensor rank. *Proceedings 43rd Annual ACM Symposium on Theory of Computing 2011*, pages 509–518, 2011. doi:10.1145/1993636.1993704.

13 Peter Bürgisser and Christian Ikenmeyer. Explicit lower bounds via geometric complexity theory. *Proceedings 45th Annual ACM Symposium on Theory of Computing 2013*, pages 141–150, 2013. doi:10.1145/2488608.2488627.

14 Peter Bürgisser, Christian Ikenmeyer, and Greta Panova. No occurrence obstructions in geometric complexity theory. *Proceedings IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 386–395, 2016. doi:10.1109/FOCS.2016.49.

15 Peter Bürgisser, Joseph M. Landsberg, Laurent Manivel, and Jerzy Weyman. An overview of mathematical issues arising in the geometric complexity theory approach to VP $\neq$ VNP. *SIAM J. Comput.*, 40(4):1179–1209, 2011. doi:10.1137/090765328.

16 Klim Efremenko, Joseph M. Landsberg, Hal Schenck, and Jerzy Weyman. The method of shifted partial derivatives cannot separate the permanent from the determinant. *ArXiv*, 2016. arXiv:1609.02103.

17 Michael Forbes. Some concrete questions on the border complexity of polynomials. Presentation given at the Workshop on Algebraic Complexity Theory WACT 2016 in Tel Aviv, https://www.youtube.com/watch?v=1HMogQIHT6Q, 2016.

18 Michael A. Forbes, Amir Shpilka, and Ben Lee Volk. Succinct hitting sets and barriers to proving algebraic circuits lower bounds. *ArXiv*, 2017. arXiv:1701.05328.

19 Fulvio Gesmundo. Geometric aspects of iterated matrix multiplication. *J. Algebra*, 461:42–64, 2016. arXiv:1512.00766, doi:10.1016/j.jalgebra.2016.04.028.

20 Joshua A. Grochow. Unifying known lower bounds via geometric complexity theory. *Comput. Complexity*, 24(2):393–475, 2015. doi:10.1007/s00037-015-0103-x.

21 Joshua A. Grochow, Ketan D. Mulmuley, and Youming Qiao. Boundaries of VP and VNP. In *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*, volume 55 of *LIPIcs*, pages 34:1–34:14, 2016. arXiv:1605.02815, doi:10.4230/LIPIcs.ICALP.2016.34.

22 Yonghui Guan. Brill's equations as a GL(V)-module. *ArXiv*, 2015. arXiv:1508.02293.

23 Ankit Gupta, Pritish Kamath, Neeraj Kayal, and Ramprasad Saptharishi. Approaching the chasm at depth four. In *2013 IEEE Conference on Computational Complexity – CCC 2013*, pages 65–73. IEEE Computer Soc., Los Alamitos, CA, 2013. doi:10.1109/CCC.2013.16.

24 Ankit Gupta, Pritish Kamath, Neeraj Kayal, and Ramprasad Saptharishi. Arithmetic circuits: a chasm at depth three. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science – FOCS 2013*, pages 578–587. IEEE Computer Soc., Los Alamitos, CA, 2013. doi:10.1109/FOCS.2013.68.

25 Johan Håstad. The shrinkage exponent of De Morgan formulas is 2. *SIAM J. Comput.*, 27(1):48–64, 1998. doi:10.1137/S0097539794261556.

26 Jonathan D. Hauenstein, Christian Ikenmeyer, and Joseph M. Landsberg. Equations for lower bounds on border rank. *Exp. Math.*, 22(4):372–383, 2013. doi:10.1080/10586458.2013.825892.

27 Pavel Hrubeš and Iddo Tzameret. Short proofs for the determinant identities. *SIAM J. Comput.*, 44(2):340–383, 2015. doi:10.1137/130917788.

28 Russell Impagliazzo and Noam Nisan. The effect of random restrictions on formula size. *Random Structures Algorithms*, 4(2):121–133, 1993. doi:10.1002/rsa.3240040202.

29 K. A. Kalorkoti. A lower bound for the formula size of rational functions. *SIAM J. Comput.*, 14(3):678–687, 1985. doi:10.1137/0214050.

30 Pascal Koiran. Arithmetic circuits: the chasm at depth four gets wider. *Theoret. Comput. Sci.*, 448:56–65, 2012. doi:10.1016/j.tcs.2012.03.041.

31 Joseph M. Landsberg. The border rank of the multiplication of 2×2 matrices is seven. *J. Amer. Math. Soc.*, 19(2):447–459, 2006. doi:10.1090/S0894-0347-05-00506-0.

32 Joseph M. Landsberg, Laurent Manivel, and Nicolas Ressayre. Hypersurfaces with degenerate duals and the geometric complexity theory program. *Comment. Math. Helv.*, 88(2):469–484, 2013. doi:10.4171/CMH/292.

33 Joseph M. Landsberg and Mateusz Michałek. A $2n^2 - \log(n) - 1$ lower bound for the border rank of matrix multiplication. *arXiv*, 2016. arXiv:1608.07486.

34 Joseph M. Landsberg and Giorgio Ottaviani. New lower bounds for the border rank of matrix multiplication. *Theory Comput.*, 11:285–298, 2015. arXiv:1112.6007, doi:10.4086/toc.2015.v011a011.

35 Thomas Lickteig. A note on border rank. *Inf. Process. Lett.*, 18(3):173–178, 1984. doi:10.1016/0020-0190(84)90023-1.

36 Guillaume Malod and Natacha Portier. Characterizing Valiant's algebraic complexity classes. *J. Complexity*, 24(1):16–38, 2008. doi:10.1016/j.jco.2006.09.006.

37 Thierry Mignon and Nicolas Ressayre. A quadratic bound for the determinant and permanent problem. *Int. Math. Res. Not.*, 2004(79):4241–4253, 2004. doi:10.1155/S1073792804142566.

38 Ketan D. Mulmuley and Milind Sohoni. Geometric complexity theory. I. An approach to the P vs. NP and related problems. *SIAM J. Comput.*, 31(2):496–526, 2001. doi:10.1137/S009753970038715X.

39 Ketan D. Mulmuley and Milind Sohoni. Geometric complexity theory II. Towards explicit obstructions for embeddings among class varieties. *SIAM J. Comput.*, 38(3):1175–1206, 2008. doi:10.1137/080718115.

40 Noam Nisan. Lower bounds for non-commutative computation. In *Proceedings of the twenty-third annual ACM symposium on Theory of computing*, pages 410–418. ACM, 1991. doi:10.1145/103418.103462.

41 Noam Nisan and Avi Wigderson. Lower bounds on arithmetic circuits via partial derivatives. *Comput. Complexity*, 6(3):217–234, 1996/97. doi:10.1007/BF01294256.

42 Luke Oeding and Steven V. Sam. Equations for the fifth secant variety of segre products of projective spaces. *Exp. Math.*, 25(1):94–99, 2016. doi:10.1080/10586458.2015.1037872.

43 Michael S. Paterson and Uri Zwick. Shrinkage of De Morgan formulae under restriction. *Random Structures Algorithms*, 4(2):135–150, 1993. doi:10.1002/rsa.3240040203.

44 Ran Raz. Multi-linear formulas for permanent and determinant are of super-polynomial size. *J. ACM*, 56(2):Art. 8, 17, 2009. doi:10.1145/1502793.1502797.

45 Herbert John Ryser. *Combinatorial mathematics*. The Carus Mathematical Monographs, No. 14. Published by The Mathematical Association of America; distributed by John Wiley and Sons, Inc., New York, 1963.

46 Chandan Saha, Ramprasad Saptharishi, and Nitin Saxena. The power of depth 2 circuits over algebras. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science*, volume 4, pages 371–382, 2009. arXiv:0904.2058, doi:10.4230/LIPIcs.FSTTCS.2009.2333.

47 Ramprasad Saptharishi. A survey of lower bounds in arithmetic circuit complexity 3.0.2, 2016. Online survey, https://github.com/dasarpmar/lowerbounds-survey. URL: https://github.com/dasarpmar/lowerbounds-survey.

48 Volker Strassen. Rank and optimal computation of generic tensors. *Linear Algebra Appl.*, 52/53:645–685, 1983. doi:10.1016/0024-3795(83)80041-X.

49 Bella Abramovna Subbotovskaya. Realizations of linear functions by formulas using +,·,-. *Doklady Akademii Nauk SSSR*, 136(3):553–555, 1961.

50 Avishay Tal. Shrinkage of De Morgan formulae by spectral techniques. In *55th Annual IEEE Symposium on Foundations of Computer Science – FOCS*, pages 551–560. IEEE Computer Soc., Los Alamitos, CA, 2014. doi:10.1109/FOCS.2014.65.

51 Sébastien Tavenas. Improved bounds for reduction to depth 4 and depth 3. *Inform. and Comput.*, 240:2–11, 2015. doi:10.1016/j.ic.2014.09.004.

52 Seinosuke Toda. Classes of arithmetic circuits capturing the complexity of computing the determinant. *IEICE Trans. Inf. & Syst.*, 75(1):116–124, 1992.

53 Leslie G. Valiant. Completeness classes in algebra. In *Conference Record of the Eleventh Annual ACM Symposium on Theory of Computing (Atlanta, Ga., 1979)*, pages 249–261. ACM, New York, 1979. doi:10.1145/800135.804419.

54 Leslie G. Valiant. *Reducibility by algebraic projections*. University of Edinburgh, Department of Computer Science, 1980. Internal Report.

55 Jeroen Zuiddam. A note on the gap between rank and border rank. *Linear Algebra Appl.*, 525:33–44, 2017. arXiv:1504.05597, doi:10.1016/j.laa.2017.03.015.

A Overview figure

The diagram in Fig. A gives an overview of inclusions and separations of complexity classes.

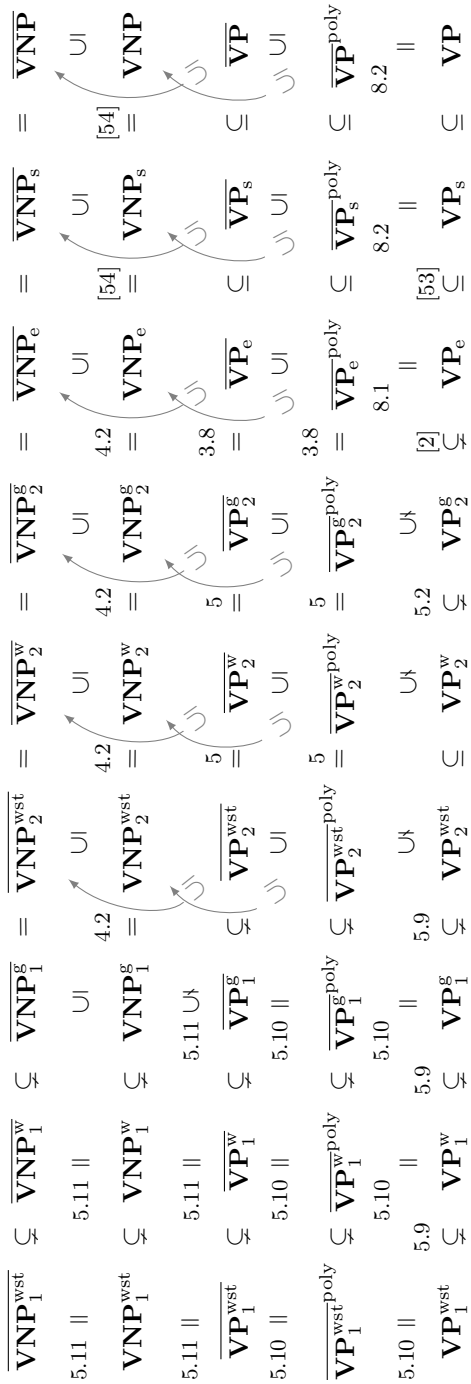


Figure A Overview of inclusions and separations among $\mathbf{VP}_k^*$, $\mathbf{VP}_{\mathrm{e}}$, $\mathbf{VP}_{\mathrm{s}}$, $\mathbf{VP}_{\mathrm{e}}$ and their closures when $\mathrm{char}(\mathbb{F}) \neq 2$.

Reconstruction of Full Rank Algebraic Branching Programs

Neeraj Kayal[1], Vineet Nair[2], Chandan Saha[3], and Sébastien Tavenas*[4]

1 **Microsoft Research India, Bengaluru, India**
neeraka@microsoft.com

2 **Indian Institute of Science, Bengaluru, India**
vineet.nair@csa.iisc.ernet.in

3 **Indian Institute of Science, Bengaluru, India**
chandan@csa.iisc.ernet.in

4 **Université Savoie Mont Blanc, CNRS, LAMA, Chambéry, France**
sebastien.tavenas@univ-smb.fr

Abstract

An algebraic branching program (ABP) $\mathtt{A}$ can be modelled as a product expression $X_1 \cdot X_2 \dots X_d$, where X_1 and X_d are $1 \times w$ and $w \times 1$ matrices respectively, and every other X_k is a $w \times w$ matrix; the entries of these matrices are linear forms in m variables over a field $\mathbb{F}$ (which we assume to be either $\mathbb{Q}$ or a field of characteristic poly(m)). The polynomial computed by $\mathtt{A}$ is the entry of the 1×1 matrix obtained from the product $\prod_{k=1}^{d} X_k$. We say $\mathtt{A}$ is a *full rank* ABP if the $w^2(d-2)+2w$ linear forms occurring in the matrices $X_1, X_2, \dots, X_d$ are $\mathbb{F}$-linearly independent. Our main result is a randomized reconstruction algorithm for full rank ABPs: Given blackbox access to an m-variate polynomial f of degree at most m, the algorithm outputs a full rank ABP computing f if such an ABP exists, or outputs 'no full rank ABP exists' (with high probability). The running time of the algorithm is polynomial in m and β, where β is the bit length of the coefficients of f. The algorithm works even if X_k is a $w_{k-1} \times w_k$ matrix (with $w_0 = w_d = 1$), and $\mathbf{w} = (w_1, \dots, w_{d-1})$ is *unknown*.

The result is obtained by designing a randomized polynomial time equivalence test for the family of iterated matrix multiplication polynomial $\mathsf{IMM}_{\mathbf{w},d}$, the $(1,1)$-th entry of a product of d rectangular symbolic matrices whose dimensions are according to $\mathbf{w} \in \mathbb{N}^{d-1}$. At its core, the algorithm exploits a connection between the *irreducible invariant subspaces* of the Lie algebra of the group of symmetries of a polynomial f that is equivalent to $\mathsf{IMM}_{\mathbf{w},d}$ and the 'layer spaces' of a full rank ABP computing f. This connection also helps determine the group of symmetries of $\mathsf{IMM}_{\mathbf{w},d}$ and show that $\mathsf{IMM}_{\mathbf{w},d}$ is characterized by its group of symmetries.

1998 ACM Subject Classification F.1.3 Complexity Measures and Classes

Keywords and phrases Circuit reconstruction, algebraic branching programs, equivalence test, iterated matrix multiplication, Lie algebra

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.21

* A part of this work was done during a postdoctoral stay in Microsoft Research India.

32nd Computational Complexity Conference (CCC 2017).
Editor: Ryan O'Donnell; Article No. 21; pp. 21:1–21:61

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

1 Introduction

1.1 Circuit reconstruction

Reconstruction of arithmetic circuits is the algebraic analogue of exact learning [5] of Boolean circuits using membership and equivalence queries. A reconstruction algorithm takes input an oracle access to an m-variate degree d polynomial f computed by a size s arithmetic circuit from some circuit class $\mathcal{C}$, and outputs an arithmetic circuit (preferably from the same class) of not too large size[1] computing f. The algorithm is allowed to make two kinds of adaptive queries to the oracle: It may ask for evaluation of f at a point $\mathbf{a} \in \mathbb{F}^m$ chosen by the algorithm (membership query). It may also form a circuit C (a hypothesis) and ask if the polynomial g, computed by C, equals f; if not, the oracle returns a point $\mathbf{b} \in \mathbb{F}^m$ such that $f(\mathbf{b}) \neq g(\mathbf{b})$ (equivalence query)[2]. The desired running time of the algorithm is polynomial in m, d, s and the bit length of the coefficients of f.

Circuit reconstruction is a natural learning problem in algebraic complexity theory and is closely related to two other fundamental problems, lower bound and polynomial identity testing. Building on the ideas in [21, 2] and [28], Volkovich [42] showed that a polynomial time reconstruction algorithm for a circuit class $\mathcal{C}$ can be used to compute an m-variate multilinear polynomial h in $2^{O(m)}$ time such that any circuit from $\mathcal{C}$ computing h has size $2^{\Omega(m)}$[3]. Also, an efficient reconstruction algorithm (that uses only membership queries) for a class of circuits automatically gives an efficient blackbox[4] identity testing algorithm for the same class. In this sense, reconstruction is a 'harder' problem than lower bound and identity testing[5]. However, if we allow reconstruction algorithms to be randomized (thereby giving them the power of identity testing) then we can hope to have efficient reconstructions even for some classes of circuits for which efficient blackbox identity testing algorithms are not known yet. Indeed, a randomized polynomial time reconstruction algorithm for read-once oblivious algebraic branching programs (ROABP) was given in [29] much before the quasi-polynomial time hitting-set generators for the same model were designed [14, 3]. The case of read-once formulas is similar (see [39]). A randomized reconstruction algorithm need not use equivalence queries as a random point $\mathbf{b}$ is a witness for $f(\mathbf{b}) \neq g(\mathbf{b})$, if $f \neq g$[6]. In this article, we will assume that reconstruction algorithms use *only* membership queries, unless we mention equivalence queries explicitly.

Another way to moderate the reconstruction setup is given by *average-case* reconstruction. Here the input polynomial f is picked according to some 'natural' distribution on circuits from a class $\mathcal{C}$. This relaxation led to the development of randomized polynomial time reconstruction algorithm for some powerful circuit classes [17, 19] (albeit on average), including arithmetic formulas for which we do not know of any super-polynomial lower bound. The notion of average-case reconstruction is related to *pseudo-random* polynomial

---

[1] We allow the algorithm to output sub-optimal size circuit as it is NP-hard to compute an optimal circuit for f even for restricted classes like set-multilinear depth three circuits [20].

[2] Throughout this article we will assume that the base field $\mathbb{F}$ is sufficiently large, so if $f(\mathbf{b}) = g(\mathbf{b})$ for every $\mathbf{b} \in \mathbb{F}^m$ then $f = g$.

[3] Such an implication is not known for an h belonging to a VNP family.

[4] The algorithm has blackbox access to f, i.e. it can make only membership queries to an oracle holding f.

[5] Not much is known about the reverse direction: Do strong lower bounds or efficient blackbox identity testing for a circuit class imply efficient reconstruction for the same class? For certain interesting circuit classes, the techniques used for identity testing and lower bounds do help in efficient reconstruction (see [39, 17]).

[6] The algorithm in [29] is deterministic if we allow equivalence queries.

families[7] and the prospects/limitations of lower bound proofs: An efficient reconstruction algorithm for polynomials generated according to a distribution D on circuits from class $\mathcal{C}$ implies that D does not generate a pseudo-random polynomial family. Such an algorithm gives evidence (contingent on the extent of naturalness of D) that most circuits in $\mathcal{C}$ have sufficient "structural/mathematical" properties in them that the reconstruction algorithm is able to exploit efficiently to distinguish polynomials computed by them from random polynomials. This *may* hint at an intriguing possibility that $\mathcal{C}$ is adequately 'weak' and amenable to explicit lower bound proofs against it. On the contrary, if D does generate a pseudo-random polynomial family then certain widely used strategies to prove lower bounds will not work for $\mathcal{C}$, much like natural proofs for Boolean circuits [36] (see the discussion in [1]).

Previous work on reconstruction

We will assume that a circuit from class $\mathcal{C}$ computing the input polynomial f has a sum gate at the output. Otherwise, we can apply the factorization algorithm in [22] to gain blackbox access to all the irreducible factors of f, thereby reducing the problem to a potentially simpler class of circuits at the cost of making the reconstruction algorithm randomized. Thus, depth two, depth three and depth four circuits would mean $\Sigma\Pi$, $\Sigma\Pi\Sigma$ and $\Sigma\Pi\Sigma\Pi$ circuits respectively.

Low depth circuits: A polynomial time reconstruction algorithm for depth two circuits follows from the sparse polynomial interpolation algorithm in [30]. By analysing the rank of the partial derivatives matrix, Klivans and Shpilka [29] gave a randomized reconstruction algorithm[8] for depth three circuits with fan-in of every product gate bounded by d in time polynomial in the size of the circuit and 2^d. Prior to this, a polynomial time randomized reconstruction algorithm for set-multilinear depth three circuits followed from [7]. In both [29] and [7] the output hypothesis is an ROABP. For depth three circuits with two product gates, Shpilka [37] gave a randomized reconstruction algorithm over a finite field $\mathbb{F}$ that has running time quasi-polynomial[9] in m, d and $|\mathbb{F}|$. This algorithm was derandomized and extended to depth three circuits with constant number of product gates in [23]. The output hypothesis in [37] is a depth three circuit with two product gates (unless the circuit has a low *simple rank*[10]), but it works only over finite fields. Recently, Sinha [40] gave a polynomial time randomized reconstruction algorithm for depth three circuits with two product gates over rationals[11]; the output of Sinha's algorithm is also a depth three circuit with two product gates (unless the simple rank of the circuit is less than a fixed constant). For multilinear depth four circuits with two top level product gates, [18] gave a randomized polynomial time reconstruction algorithm that works over both finite fields and rationals.

Restricted formulas and ABP: Recently, Minahan and Volkovich [33] gave a polynomial time reconstruction algorithm for read-once formulas by strengthening the analysis in [38], the

[7] Intuitively, a distribution D on m-variate degree-d polynomials using a random seed of length $s = (md)^{O(1)}$ generates a pseudo-random polynomial family if any algorithm that distinguishes polynomials coming from D from uniformly-random m-variate degree-d polynomials with a non-negligible bias, takes time exponential in s.

[8] The algorithm is deterministic if equivalence queries are used.

[9] The running time is polynomial in $m, |\mathbb{F}|$ if the depth three circuit is additionally multilinear.

[10] The dimension of the span of the linear forms in the two gates after removing their gcd.

[11] The result holds over characteristic zero fields. We state it for rationals as bit complexity concerns us.

latter has a quasi-polynomial time reconstruction algorithm for the same model. Forbes and Shpilka [14] gave quasi-polynomial time reconstruction algorithms for ROABP, set-multilinear ABP and non-commutative ABP by derandomizing[12] the algorithm in [29]. Prior to this, the case of non-commutative ABP reconstruction was solved in [6] assuming blackbox access to the internal gates of the input ABP.

Average-case reconstruction: Few reconstruction algorithms are known under distributional assumptions on the inputs. Gupta, Kayal and Lokam [17] gave a randomized polynomial time reconstruction algorithm for random multilinear formulas picked from a natural distribution: every sum gate computes a random linear combinations of its two children (subformulas), and at every product gate the set of variables is partitioned randomly into two equal size sets between its two children (subformulas); the subformulas are then constructed recursively. In [19], a randomized polynomial time reconstruction algorithm was given for random formulas picked from the distribution of size s complete binary trees with alternating layers of sum and product gates, and the linear forms at the leaves are chosen independently and uniformly at random.

1.2 Motivation and model

Motivation: Given the results in [17, 19], it is natural to study the complexity of average-case reconstruction for models more powerful than formulas, like ABPs. Another motivation is the following: Aaronson [1] gave a candidate for pseudo-random family of low degree polynomials over a finite field $\mathbb{F}$. There it is conjectured that the family $\{\mathsf{Det}_d(A \cdot \mathbf{x}) : A \in \mathbb{F}^{d^2 \times m}\}$, where Det_d is the determinant of a $d \times d$ symbolic matrix and $|\mathbf{x}| = m$, is pseudo-random if A is chosen uniformly at random from $\mathbb{F}^{d^2 \times m}$ and $m \ll d$. If this is shown to be true (under *plausible* hardness assumptions) then that would demonstrate a natural-proofs-like barrier in the algebraic world. Although the conjecture is made for finite fields, it remains interesting to study even if the entries of A are chosen from a large enough subset of $\mathbb{Q}$ (or $\mathrm{char}(\mathbb{F}) > d^c$ for a sufficiently large constant c). Moreover, since determinant is complete (under p-projections) for algebraic branching programs [32] and so is $\mathsf{IMM}_{w,d}$ – the $(1,1)$-th entry of a product of d $w \times w$ symbolic matrices – it is natural to ask if $\{\mathsf{IMM}_{w,d}(A \cdot \mathbf{x}) : A \in \mathbb{F}^{n \times m}\}$ is also a pseudo-random polynomial family when A is random and $m \ll w^2 d$; here $n = w^2(d-2) + 2w$ is the number of variables in $\mathsf{IMM}_{w,d}$. If yes then we cannot hope to design an efficient reconstruction algorithm for algebraic branching programs in the average-case. On the other hand, if such an average-case reconstruction is possible then the above family generated by linear projections of $\mathsf{IMM}_{w,d}$ is not pseudo-random. This motivates us to pose Problem 3 below (rather optimistically), and study a couple of special cases when it can be solved – one is in this article and the other in an upcoming work [27][13].

Algebraic branching program: Algebraic branching program (ABP), an arithmetic analogue of Boolean branching program, is a well-studied model in algebraic complexity theory specially because it captures the complexity of polynomials like the iterated matrix multiplication and the symbolic determinant. Separating the computational powers of formulas and ABPs, and that of ABPs and circuits are outstanding open problems in arithmetic circuit complexity.

[12] Replacing the equivalence queries by quasi-polynomial size hitting-sets for ROABP.
[13] See Section 1.4 for some details on this work.

An ABP is defined below. For the rest of this article, the base field $\mathbb{F}$ would be the field of rationals $\mathbb{Q}$[14].

▶ **Definition 1** (Algebraic branching program)**.** An *algebraic branching program* (ABP) of width w and length d is a product expression $X_1 \cdot X_2 \dots X_d$, where X_1, X_d are row and column vectors of length w respectively, and for $k \in [2, d-1]$, X_k is a $w \times w$ matrix. The entries in X_1 to X_d are affine forms in the variables $\mathbf{x} = \{x_1, x_2, \dots, x_m\}$. The polynomial computed by the ABP is the entry of the 1×1 matrix obtained from the product $\prod_{k=1}^{d} X_k$. An ABP of width w, length d, and in m variables, and with the coefficients of the affine forms from $S \subseteq \mathbb{F}$, will be called a (w, d, m, S)-ABP.

An alternate definition: Alternatively, an ABP is defined as a layered directed acyclic graph with a source s and a sink t. A width w and length d ABP has $d+1$ layers, where the first and the last layers contain one vertex each, labelled s and t respectively, and every other layer has w vertices. There is an edge from every vertex in layer k to every vertex in layer $k+1$, for all $k \in [d]$, and these edges between adjacent layers are labelled by affine forms in $\mathbf{x}$ variables. The weight of a path from s to t is the product of the edge labels in the path, and the polynomial computed by the ABP is the sum of the weights of all paths from s to t. It is easy to verify that the two definitions of ABP are equivalent. We use either of these definitions in our arguments later based on suitability.

Average-case ABP reconstruction: In order to study average-case complexity of the reconstruction problem for ABPs, we need to define a distribution on polynomials computed by ABPs. A seemingly natural distribution is as follows: Consider the universe of all polynomials computed by (w, d, m, S)-ABPs for some finite set $S \subseteq \mathbb{F}$ of large enough size. Pick a polynomial f uniformly at random from this universe, and give blackbox access to f as input to a reconstruction algorithm. However, a distribution is 'realistic' only if there is an efficient sampling algorithm that outputs (some suitable circuit representation of) f according to the distribution. For the above distribution, it is not clear if such an efficient sampling algorithm exists. A reason being, multiple different ABPs may be computing the same polynomial, so picking a random ABP is not sufficient to sample from this distribution. However, picking a random ABP (as described below) gives another natural distribution for which there is a trivial efficient sampling algorithm. Let S_γ be the set of all positive and negative rational numbers with γ bits before and after the decimal.

▶ **Definition 2** (Random algebraic branching program)**.** Given the parameters w, d, m and γ, a *random* (w, d, m, S_γ)-ABP is a (w, d, m, S_γ)-ABP with coefficients of the affine forms chosen independently and uniformly at random from S_γ[15].

Indeed there is a randomized sampling algorithm which when given the parameters w, d, m and γ outputs a random (w, d, m, S_γ)-ABP in time $(w, d, m, \gamma)^{O(1)}$. An average-case ABP reconstruction problem can then be posed as follows.

▶ **Problem 3** (Average-case ABP reconstruction)**.** *Design an algorithm which when given blackbox access to a polynomial f computed by a random (w, d, m, S_γ)-ABP, outputs an*

[14] Our results also hold over finite fields of sufficiently large (meaning, polynomial in the relevant parameters) characteristic.

[15] More generally, S_γ can be any arbitrarily fixed set containing rational numbers of the form $\frac{p}{q}$, where p and q are γ bit integers. For concreteness of the discussion we have fixed S_γ in a specific way.

ABP computing f with high probability[16] *. The desired running time of the algorithm is* $(w, d, m, \gamma)^{O(1)}$.

Note that we allow the reconstruction algorithm to output any ABP computing f which may not be a (w, d, m, S_γ)-ABP. The main requirement is that the running time should be polynomial in w, d, m and γ.

1.3 Our result

We give a solution to the above problem, if the number of variables m and the size of the set S_γ are greater than $w^2 d$ and $(mwd)^2$ respectively. Observe that if the random affine forms in the matrices X_1 to X_d (as in Definition 2) have more than $w^2 d$ variables then these affine forms are $\mathbb{F}$-linearly independent with high probability as S_γ is also sufficiently large. This motivates us to define a *full rank* ABP. In the following discussion, by *homogeneous degree 1 part* of an affine form $a_0 + \sum_{i=1}^{m} a_i x_i$ we mean $\sum_{i=1}^{m} a_i x_i$ where $a_i \in \mathbb{F}$.

▶ **Definition 4** (Full rank algebraic branching program)**.** A *full rank* ABP $\mathtt{A}$ of width bounded by w and length d is a product expression $X_1 \cdot X_2 \dots X_d$, where X_1, X_2 are row and column vectors of lengths w_1 and w_{d-1} respectively, and for $k \in [2, d-1]$ X_k is a $w_{k-1} \times w_k$ matrix such that $w_k \leq w$ for all $k \in [d-1]$; the entries in X_1 to X_d are affine forms in $\mathbf{x}$ variables and moreover, the homogeneous degree 1 parts of these affine forms are $\mathbb{F}$-linearly independent. We say ABP $\mathtt{A}$ has *width* $\mathbf{w} = (w_1, w_2, \dots, w_{d-1}) \in \mathbb{N}^{d-1}$.

The following is an example of a full rank ABP,

$$\begin{bmatrix} 1 + x_1 + x_2 & 2 + x_2 + x_3 & x_3 + x_4 \end{bmatrix} \begin{bmatrix} 1 + x_4 + x_5 & x_5 + x_6 \\ x_6 + x_7 & x_7 + x_8 \\ x_8 + x_9 & 4 + x_9 + x_{10} \end{bmatrix} \begin{bmatrix} 3 + x_{10} + x_{11} \\ 2 + x_{11} \end{bmatrix}.$$

A canonical example: Another example of a polynomial computed by a full rank ABP is the iterated matrix multiplication polynomial $\mathsf{IMM}_{\mathbf{w},d}$, which is the entry of the 1×1 matrix obtained from a product of d *symbolic* matrices X_1 to X_d with dimensions as in Definition 4. The number of variables in $\mathsf{IMM}_{\mathbf{w},d}$ is $n = w_1 + \sum_{k=2}^{d-1} w_{k-1} w_k + w_{d-1}$. See Definition 2.3 for a slightly detailed definition of $\mathsf{IMM}_{\mathbf{w},d}$. Generally in the literature, the matrices X_1 to X_d have a uniform dimension w (i.e. $w_k = w$ for every $k \in [d-1]$) and the polynomial is denoted by $\mathsf{IMM}_{w,d}$. We consider varying dimensions primarily because the algorithm in Theorem 5 below is able to handle this general setting, even if $\mathbf{w}$ is *unknown*.

Our main result is an efficient randomized algorithm to reconstruct full rank ABP.

▶ **Theorem 5** (Full rank ABP reconstruction)**.** *There is a randomized algorithm that takes as input a blackbox for an m variate polynomial f over $\mathbb{F}$ of degree $d \in [5, m]$, and with high probability it does the following: if f is computed by a full rank ABP then the algorithm outputs a full rank ABP computing f, else it outputs 'f does not admit a full rank ABP'. The running time is* $\mathrm{poly}(m, \beta)$[17]*, where β is the bit length of the coefficients of f.*

[16] The probability is taken over the random choice of f (the polynomial computed by a random (w, d, m, S_γ)-ABP) as well as over the random bits used by the reconstruction algorithm, if it is randomized.

[17] Throughout this article $\mathrm{poly}(m)$ denotes a sufficiently large polynomial function in m; $\mathrm{poly}(m, \beta)$ is defined similarly.

Remarks: Theorem 5 implies an efficient average-case reconstruction algorithm for ABPs (Problem 3) when $m \geq w^2 d$ and $|S_\gamma| \geq (mwd)^2$, as a random (w, d, m, S_γ)-ABP is full rank with high probability if m and $|S_\gamma|$ are sufficiently large. The algorithm of Theorem 5 is given in Section 1.5. Following are a couple of remarks on this algorithm:

1. *Uniqueness of full rank ABP*: Suppose f is computed by a full rank ABP of width $\mathbf{w} = (w_1, w_2, \ldots, w_{d-1})$, and assume[18] that $w_k > 1$ for every $k \in [d-1]$. Then the output of the algorithm is a full rank ABP of width $\mathbf{w}$ or $(w_{d-1}, w_{d-2}, \ldots, w_1)$, with probability at least $1 - \frac{1}{\text{poly}(w,d)}$, where $w = \max_{k\in[d-1]}\{w_k\}$. In fact, any full rank ABP computing f is 'unique' up to the symmetries[19] of iterated matrix multiplication which we study in Section 6.
2. *No knowledge of* $\mathbf{w}$: The algorithm does not need a priori knowledge of the width vector $\mathbf{w}$, it only knows the number of variables m and the degree d of f. The algorithm is able to derive $\mathbf{w}$ from blackbox access to f (Section 1.5 gives a sketch of how this is done).

Observe that if f is computed by a full rank ABP of width $\mathbf{w}$ then f is an affine projection of the polynomial $\mathsf{IMM}_{\mathbf{w},d}$ via a full rank transformation (see Definition 17). So the above theorem is identical to the theorem below.

▶ **Theorem 6.** *Given blackbox access to an m variate polynomial $f \in \mathbb{F}[\mathbf{x}]$ of degree $d \in [5, m]$, the problem of checking if there exist a $\mathbf{w} \in \mathbb{N}^{d-1}$, a $B \in \mathbb{F}^{n\times m}$ of rank n equal to the number of variables in $\mathsf{IMM}_{\mathbf{w},d}$, and a $\mathbf{b} \in \mathbb{F}^n$ such that $f = \mathsf{IMM}_{\mathbf{w},d}(B\mathbf{x} + \mathbf{b})$[20], can be solved in randomized $\text{poly}(m, \beta)$ time where β is the bit length of the coefficients of f. Further, with probability at least $1 - \frac{1}{\text{poly}(n)}$, the following is true: the algorithm returns a $\mathbf{w}$, a $B \in \mathbb{F}^{n\times m}$ of rank n, and a $\mathbf{b} \in \mathbb{F}^n$ such that $f = \mathsf{IMM}_{\mathbf{w},d}(B\mathbf{x} + \mathbf{b})$ if such $\mathbf{w}$, B and $\mathbf{b}$ exist, else it outputs 'f does not admit a full rank ABP'.*

A full rank ABP for f can be derived readily, once we compute $\mathbf{w}$, B and $\mathbf{b}$ as above. Using known results on variable reduction and translation equivalence test (see Section 2.2) proving Theorem 6 reduces in polynomial time to giving an equivalence test (see Definition 18) for the $\mathsf{IMM}_{\mathbf{w},d}$ polynomial – this reduction is described in Section 1.5.

▶ **Theorem 7** (Equivalence test for IMM)**.** *Given blackbox access to a homogeneous n variate polynomial $f \in \mathbb{F}[\mathbf{x}]$ of degree $d \in [5, n]$, where $|\mathbf{x}| = n$, the problem of checking if there exist a $\mathbf{w} \in \mathbb{N}^{d-1}$ and an invertible $A \in \mathbb{F}^{n\times n}$ such that $f = \mathsf{IMM}_{\mathbf{w},d}(A\mathbf{x})$, can be solved in randomized $\text{poly}(n, \beta)$ time where β is the bit length of the coefficients of f. Further, with probability at least $1 - \frac{1}{\text{poly}(n)}$ the following holds: the algorithm returns a $\mathbf{w}$, and an invertible $A \in \mathbb{F}^{n\times n}$ such that $f = \mathsf{IMM}_{\mathbf{w},d}(A\mathbf{x})$ if such $\mathbf{w}$ and A exist, else it outputs 'no such $\mathbf{w}$ and A exist'.*

Remarks: Suppose $f = \mathsf{IMM}_{\mathbf{w},d}(A\mathbf{x})$ for an invertible $A \in \mathbb{F}^{n\times n}$ and $\mathbf{w} = (w_1, w_2, \ldots, w_{d-1})$.

1. *Irreducibility of* $\mathsf{IMM}_{\mathbf{w},d}$: We can assume without loss of generality that $w_k > 1$ for every $k \in [d-1]$, implying $\mathsf{IMM}_{\mathbf{w},d}$ is an irreducible polynomial. If $w_k = 1$ for some $k \in [d-1]$ then $\mathsf{IMM}_{\mathbf{w},d}$ is reducible, in which case we use the factorization algorithm in [22] to get blackbox access to the irreducible factors of f and then apply Theorem 7 to each of these irreducible factors (Section 1.5 has more details on this).

[18] The first remark after Theorem 7 justifies this assumption.
[19] The stabilizer under the action of the general linear group.
[20] A variable set $\mathbf{x} = \{x_1, \ldots, x_m\}$ is treated as a column vector $(x_1 \ldots x_m)^T$ in the expression $B\mathbf{x} + \mathbf{b}$. The affine form entries of the column $B\mathbf{x} + \mathbf{b}$ are then plugged in place of the variables of $\mathsf{IMM}_{\mathbf{w},d}$ (following a variable ordering, like the one mentioned in Section 2.3).

2. *Uniqueness of* $\mathbf{w}$ *and* A: Assuming $w_k > 1$ for every $k \in [d-1]$, it would follow from the proof of the theorem that $\mathbf{w}$ is unique in the following sense: if $f = \mathsf{IMM}_{\mathbf{w}',d}(A'\mathbf{x})$, where $A' \in \mathbb{F}^{n\times n}$ is invertible, then either $\mathbf{w}' = \mathbf{w}$ or $\mathbf{w}' = (w_{d-1}, w_{d-2}, \ldots, w_1)$. Since $f = X_1 \cdot X_2 \ldots X_d = X_d^T \cdot X_{d-1}^T \ldots X_1^T$, $\mathbf{w}'$ can indeed be $(w_{d-1}, w_{d-2}, \ldots, w_1)$. The invertible transformation A is also unique up to the group of symmetries (see Defintion 19) of $\mathsf{IMM}_{\mathbf{w},d}$: if $\mathsf{IMM}_{\mathbf{w},d}(A\mathbf{x}) = \mathsf{IMM}_{\mathbf{w},d}(A'\mathbf{x})$ then AA'^{-1} is in the group of symmetries of $\mathsf{IMM}_{\mathbf{w},d}$. In Section 6, we determine this group and show that $\mathsf{IMM}_{\mathbf{w},d}$ is characterized by it.
3. *A related result in [16]*: Another useful definition of the iterated matrix multiplication polynomial is the trace of a product of d $w \times w$ symbolic matrices – let us denote this polynomial by $\mathsf{IMM}'_{w,d}$. Both the variants, $\mathsf{IMM}'_{w,d}$ and $\mathsf{IMM}_{w,d}$, are well-studied in the literature and their circuit complexities are polynomially related. However, an equivalence test for one does not immediately give an equivalence test for the other. This is partly because the group of symmetries of $\mathsf{IMM}'_{w,d}$ and $\mathsf{IMM}_{w,d}$ are not exactly the same in nature (see Section 6 for a comparison).
   Let $\mathbf{x}_1, \ldots, \mathbf{x}_d$ be the sets of variables in the d matrices of $\mathsf{IMM}'_{w,d}$ respectively. A polynomial $f(\mathbf{x}_1, \ldots, \mathbf{x}_d)$ is said to be *multilinearly equivalent* to $\mathsf{IMM}'_{w,d}$ if there are invertible $w \times w$ matrices $A_1, \ldots, A_d$ such that $f = \mathsf{IMM}'_{w,d}(A_1\mathbf{x}_1, \ldots, A_d\mathbf{x}_d)$. Grochow [16] showed the following result: Given the knowledge of the variable sets $\mathbf{x}_1, \ldots, \mathbf{x}_d$, an oracle to find roots of univariate polynomials over $\mathbb{C}$ and blackbox access to a polynomial f, there is a randomized algorithm to check whether f is multilinearly equivalent to $\mathsf{IMM}'_{w,d}$ using $\mathrm{poly}(w, d)$ operations over $\mathbb{C}$. Due to the issue of representing complex numbers, the model of computation for this result may be assumed to be the Blum-Shub-Smale model [10]. Theorem 7 is different from the result in [16] in a few ways: First, the equivalence test is for $\mathsf{IMM}_{\mathbf{w},d}$ instead of $\mathsf{IMM}'_{w,d}$. The algorithm in Theorem 7 operates without the knowledge of the variable sets $\mathbf{x}_1, \ldots, \mathbf{x}_d$ (in fact, without the knowledge of $\mathbf{w}$). It only "sees" n variables $x_1, \ldots, x_n$ that are input to the blackbox for f. Second, there is no requirement of a oracle for finding roots of univariates. The base field is $\mathbb{Q}$ or a field with sufficiently large characteristic and the model of computation is the Turing machine model. Third, Theorem 7 gives a general equivalence test whereas the algorithm in [16] checks only multilinear equivalence.

1.4 Discussion

To summarize, our main contribution is a polynomial time randomized equivalence test for $\mathsf{IMM}_{\mathbf{w},d}$, even if $\mathbf{w}$ is unknown. Although, equivalence testing is an important problem in its own right, Theorem 5 does not address the average-case ABP reconstruction problem quite satisfactorily because of the restriction $m \geq w^2 d$[21]. Keeping the conjecture [1] on pseudo-random polynomial family in mind, the more interesting and challenging scenario is when $m \ll w^2 d$ in Problem 3, and this case remains an open problem. We address this problem partially in an upcoming work (and equivalence tests feature in there too).

A forthcoming work [27]: If the width w of the ABP is a constant, we still need $m = \Omega(d)$, for a random ABP to have full rank and Theorem 5 to be effective. The case of constant width ABP is interesting in its own right as they capture the complexity of arithmetic

[21] Besides, the model full rank ABP, although natural and powerful, is nevertheless incomplete – not every polynomial f can be computed by a full rank ABP even if f is multilinear (see Observation 60).

formulas. In particular, if a polynomial g is computed by a formula of size s then g can be computed as the $(1,1)$-th entry of a product of $s^{O(1)}$ many 3×3 matrices with affine form entries [8], and every polynomial computed by a size s width 3 ABP can be computed by a formula of size $s^{O(1)}$. With constant width ABP in mind, we study a version of Problem 3 (Problem 8 below) in [27], and make progress in certain cases (particularly for $w = 3$) under the restriction $m \geq w^2$; that is for constant width, m only needs to be larger than a constant. Problem 8 is also a natural matrix factorization problem.

▶ **Problem 8** (Average-case matrix factorization). *Design an algorithm which when given a $d \in \mathbb{N}$ and blackbox access to w^2 entries of a matrix $F = X_1 \cdot X_2 \dots X_d$, where $X_1, X_2, \dots, X_d$ are $w \times w$ matrices having entries affine forms in m variables with coefficients chosen independently and uniformly at random from S_γ, computes d $w \times w$ matrices $Y_1, Y_2, \dots, Y_d$ with affine form entries such that $F = Y_1 \cdot Y_2 \dots Y_d$. The desired running time of the algorithm is* $\text{poly}(m, w, d, \gamma)$.

As before, we allow the coefficients of the affine forms in $Y_1, Y_2, \dots, Y_d$ to not belong to S_γ.

In a certain sense, Problem 8 is a relaxed version of Problem 3: We have blackbox access to all the w^2 polynomials occurring as entries of the matrix product in Problem 8, whereas in Problem 3 we have blackbox access to just a single polynomial which can be thought of as one entry of a matrix product. Nevertheless, if the coefficients of the affine forms in $X_1, X_2, \dots, X_d$ are adversarially chosen in Problem 8 (instead of independently and uniformly at random from S_γ) then the problem becomes as hard as worst-case formula reconstruction (by [8]), and this makes the above average-case variant interesting to study.

1.5 Algorithm and proof strategy

An algorithm for reconstructing full rank ABP is given in Algorithm 1. At first, we trace the steps of this algorithm to show that proving Theorem 6 reduces to proving Theorem 7 using known methods. Then, we give an equivalence test for $\mathsf{IMM}_{\mathbf{w},d}$ in Algorithm 2, which is the contribution of this work. Some relevant definitions, notations and concepts can be found in Section 2.

1.5.1 Reduction to equivalence test for IMM

We are given blackbox access to an m variate polynomial $f(\tilde{\mathbf{x}})$ in Algorithm 1 where $\tilde{\mathbf{x}} = \{x_1, \dots, x_m\}$. Suppose $f = \mathsf{IMM}_{\mathbf{w}',d}(B'\tilde{\mathbf{x}} + \mathbf{b}')$ for some unknown $\mathbf{w}' \in \mathbb{N}^{d-1}$, $\mathbf{b}' \in \mathbb{F}^n$ and $B' \in \mathbb{F}^{n \times m}$ of rank n, where n is the number of variables in $\mathsf{IMM}_{\mathbf{w}',d}$.

Variable reduction (Step 2): The number of essential/redundant variables of a polynomial remains unchanged under affine projection via full rank transformation. Since $\mathsf{IMM}_{\mathbf{w}',d}$ has no redundant variables[22], the number of essential variables of f equals n. The algorithm eliminates the $m - n$ redundant variables in f by applying Algorithm 8 and constructs a $C \in \mathsf{GL}(m)$ such that $g = f(C\tilde{\mathbf{x}})$ has only the essential variables $\mathbf{x} = \{x_1, \dots, x_n\}$. It follows that $g = \mathsf{IMM}_{\mathbf{w}',d}(A'\mathbf{x} + \mathbf{b}')$, where $A' \in \mathsf{GL}(n)$ is the matrix $B' \cdot C$ restricted to the first n columns.

[22] Which follows easily from Claim 26.

Equivalence test (Steps 5–9): Since $g = \mathsf{IMM}_{\mathbf{w}',d}(A'\mathbf{x} + \mathbf{b}')$, its d-th homogeneous component $g^{[d]} = \mathsf{IMM}_{\mathbf{w}',d}(A'\mathbf{x})$. In other words, $g^{[d]}$ is equivalent to $\mathsf{IMM}_{\mathbf{w}',d}$ for an unknown $\mathbf{w}' \in \mathbb{N}^{d-1}$. At this point, the algorithm calls Algorithm 2 to find a $\mathbf{w}$ and an $A \in \mathsf{GL}(n)$ such that $g^{[d]} = \mathsf{IMM}_{\mathbf{w},d}(A\mathbf{x})$, and this is achieved with high probability.

Finding a translation (Steps 12–17): As $g = \mathsf{IMM}_{\mathbf{w}',d}(A' \cdot (\mathbf{x} + A'^{-1}\mathbf{b}')) = g^{[d]}(\mathbf{x} + A'^{-1}\mathbf{b}')$, g is translation equivalent to $g^{[d]}$. With high probability, Algorithm 9 finds an $\mathbf{a} \in \mathbb{F}^n$ such that $g = g^{[d]}(\mathbf{x} + \mathbf{a})$, implying $g = \mathsf{IMM}_{\mathbf{w},d}(A\mathbf{x} + A\mathbf{a})$. Thus $\mathbf{b} = A\mathbf{a}$ is a valid translation vector.

Final reconstruction (Steps 20–26): From the previous steps, we have $g = \mathsf{IMM}_{\mathbf{w},d}(A\mathbf{x}+\mathbf{b})$. Although the variables $\{x_{n+1}, \ldots, x_m\}$ are absent in g, if we pretend that g is a polynomial in all the $\tilde{\mathbf{x}}$ variables then $g = \mathsf{IMM}_{\mathbf{w},d}(A_0\tilde{\mathbf{x}} + \mathbf{b})$, where A_0 is an $n \times m$ matrix such that the $n \times n$ submatrix formed by restricting to the first n columns of A_0 equals A and the remaining $m - n$ columns of A_0 have all zero entries. Hence $f = g(C^{-1}\tilde{\mathbf{x}}) = \mathsf{IMM}_{\mathbf{w},d}(A_0C^{-1}\tilde{\mathbf{x}} + \mathbf{b})$ which explains the setting $B = A_0C^{-1}$ in step 20. The identity testing in steps 21-23 takes care of the situation when, to begin with, there are no $\mathbf{w}' \in \mathbb{N}^{d-1}$, $\mathbf{b}' \in \mathbb{F}^n$ and $B' \in \mathbb{F}^{n\times m}$ of rank n such that $f = \mathsf{IMM}_{\mathbf{w}',d}(B'\tilde{\mathbf{x}} + \mathbf{b}')$.

1.5.2 Equivalence test for IMM

Algorithm 1 calls Algorithm 2 on a blackbox holding a homogeneous n variate polynomial $f(\mathbf{x})$ of degree $d \leq n$, and expects a $\mathbf{w} \in \mathbb{N}^{d-1}$ and an $A \in \mathsf{GL}(n)$ in return such that $f = \mathsf{IMM}_{\mathbf{w},d}(A\mathbf{x})$, if such $\mathbf{w}$ and A exist. First, we argue that f can be assumed to be an irreducible polynomial.

Assuming irreducibility of input f in Algorithm 2: The idea is to construct blackbox access to the irreducible factors of f using the efficient randomized polynomial factorization algorithm in [22], and compute full rank ABP for each of these irreducible factors. The ABPs are then connected 'in series' to form a full rank ABP for f. This process succeeds with high probability. The details are as follows: If f is not square-free (which can be easily checked using [22]) then f cannot be equivalent to $\mathsf{IMM}_{\mathbf{w},d}$ for any $\mathbf{w}$, as $\mathsf{IMM}_{\mathbf{w},d}$ is always square-free. Suppose $f = f_1 \cdots f_k$, where $f_1, \ldots, f_k$ are distinct irreducible factors of f. If there are $\mathbf{w}' \in \mathbb{N}^{d-1}$ and $A' \in \mathsf{GL}(n)$ such that $f = \mathsf{IMM}_{\mathbf{w}',d}(A'\mathbf{x})$, then the number of essential variables in f is n (as $\mathsf{IMM}_{\mathbf{w}',d}$ has no redundant variables). Also, $f_1 \cdots f_k = h_1(A'\mathbf{x}) \cdots h_k(A'\mathbf{x})$ where $h_1, \ldots, h_k$ are the irreducible factors of $\mathsf{IMM}_{\mathbf{w}',d}$. The irreducible factors of $\mathsf{IMM}_{\mathbf{w}',d}$ are 'smaller $\mathsf{IMM}s$' in disjoint sets of variables[23]. Hence, by uniqueness of factorization, f_ℓ is computable by a full rank ABP for every $\ell \in [k]$. Let the degree of f_ℓ be d_ℓ and n_ℓ the number of essential variables in f_ℓ. Then $n_1 + \ldots + n_k = n$. Now observe that if we invoke Algorithm 1 on input f_ℓ, it calls Algorithm 2 from within on an irreducible polynomial, as f_ℓ is homogeneous and irreducible. Algorithm 1 returns a $\mathbf{w}_\ell \in \mathbb{N}^{d_\ell - 1}$ and $B_\ell \in \mathbb{F}^{n_\ell \times n}$ of rank n_ℓ such that $f_\ell = \mathsf{IMM}_{\mathbf{w}_\ell,d_\ell}(B_\ell\mathbf{x})$ (ignoring the translation vector as f_ℓ is homogeneous). Let $\mathbf{w} \in \mathbb{N}^{d-1}$ be the vector $(\mathbf{w}_1\ 1\ \mathbf{w}_2\ 1 \ldots\ 1\ \mathbf{w}_k)$[24], and $A \in \mathbb{F}^{n\times n}$ such that the first n_1 rows of A is B_1, next n_2 rows is B_2, and so on till last n_k rows is B_k. Then, $f = \mathsf{IMM}_{\mathbf{w},d}(A\mathbf{x})$. Clearly, A

[23] Recall, $\mathsf{IMM}_{\mathbf{w},d}$ is irreducible if $w_k > 1$ for every $k \in [d-1]$ where $\mathbf{w} = (w_1, \ldots, w_{d-1})$.

[24] The notation means the entries of $\mathbf{w}_1$ are followed by 1, followed by the entries of $\mathbf{w}_2$, then a 1 again, and so on.

Algorithm 1 Reconstructing a full rank ABP

INPUT: Blackbox access to an m variate polynomial $f(\tilde{\mathbf{x}})$ of degree $d \leq m$.
OUTPUT: A full rank ABP computing f if such an ABP exists.

/* Variable reduction */
Use Algorithm 8 to compute n and $C \in \mathsf{GL}(m)$ such that $g = f(C\tilde{\mathbf{x}})$ has only the essential variables $\mathbf{x} = \{x_1, \ldots, x_n\}$ of f. If $d > n$, output 'f does not admit a full rank ABP' and stop.

/* Equivalence test: Finding $\mathbf{w}$ and A */
Construct a blackbox for $g^{[d]}$, the d-th homogeneous component of g (see Section 2.2).
Use Algorithm 2 to find a $\mathbf{w} \in \mathbb{N}^{d-1}$ and an $A \in \mathsf{GL}(n)$ such that $g^{[d]} = \mathsf{IMM}_{\mathbf{w},d}(A\mathbf{x})$.
if Algorithm 2 outputs 'no such $\mathbf{w}$ and A exist' **then**
    Output 'f does not admit a full rank ABP' and stop.
end if

/* Finding a translation $\mathbf{b}$ */
Use Algorithm 9 to find an $\mathbf{a} \in \mathbb{F}^n$ such that $g = g^{[d]}(\mathbf{x} + \mathbf{a})$.
if Algorithm 9 outputs 'g is not translation equivalent to $g^{[d]}$' **then**
    Output 'f does not admit a full rank ABP' and stop.
else
    Set $\mathbf{b} = A\mathbf{a}$.
end if

/* Identity testing and final reconstruction */
Let A_0 be the $n \times m$ matrix obtained by attaching $m - n$ 'all-zero' columns to the right of A. Set $B = A_0 C^{-1}$.
Choose a point $\mathbf{a} \in S^m$ at random, where $S \subseteq \mathbb{F}$ and $|S| \geq \text{poly}(n)$.
if $f(\mathbf{a}) \neq \mathsf{IMM}_{\mathbf{w},d}(B\mathbf{a} + \mathbf{b})$ **then**
    Output 'f does not admit a full rank ABP' and stop.
else
    Construct a full rank ABP $\mathtt{A}$ of width $\mathbf{w}$ from B and $\mathbf{b}$. Output $\mathtt{A}$.
end if

must be in $\mathsf{GL}(n)$ as the number of essential variables of f is n. Thus, it is sufficient to describe Algorithm 2 on an input f that is irreducible.

A comparison with [25] and our proof strategy: Kayal [25] gave equivalence tests for the permanent and determinant polynomials by making use of their Lie algebra (see Definition 20). Algorithm 2 also involves Lie algebra of IMM, but there are some crucial differences in the way Lie algebra is used in [25] and in here. The Lie algebra of permanent consists of diagonal matrices and hence commutative. By diagonalizing a basis of $\mathfrak{g}_f$ over $\mathbb{C}$, for an f equivalent to permanent, we can reduce the problem to the much simpler permutation and scaling (PS) equivalence problem. The Lie algebra of $n \times n$ determinant, which is isomorphic to $\mathfrak{sl}_n \oplus \mathfrak{sl}_n$, is not commutative. However, a Cartan subalgebra of $\mathfrak{sl}_n$ consists of traceless diagonal matrices. This then helps reduce the problem to PS-equivalence by diagonalizing (over $\mathbb{C}$) a basis of the centralizer of a random element in $\mathfrak{g}_f$, for an f equivalent to determinant. Both the equivalence tests involve simultaneous diagonalization of matrices over $\mathbb{C}$. It is a bit unclear

how to carry through this step if the base field is $\mathbb{Q}$ and we insist on low bit complexity. The Lie algebra of IMM is not commutative. Also, we do not know if going to Cartan subalgebra helps, as we would like to avoid the simultaneous diagonalization step. Instead of Cartan subalgebras, we study invariant subspaces (Definition 12) of the Lie algebra $\mathfrak{g}_{\mathsf{IMM}}$. A detailed analysis of the Lie algebra (in Section 3) reveals the structure of the irreducible invariant subspaces of $\mathfrak{g}_{\mathsf{IMM}}$. It is observed that these invariant subspaces are intimately connected to the layer spaces (see Definition 15) of any full rank ABP computing f. At a conceptual level, this connection helps us reconstruct a full rank ABP. Once we have access to the layer spaces, we can retrieve the unknown width vector $\mathbf{w}$ whence the problem reduces to the easier problem of reconstructing an almost set-multilinear ABP (Definition 29).

We now give some more details on Algorithm 2. Suppose there is a $\mathbf{w} \in \mathbb{N}^{d-1}$ such that f is equivalent to $\mathsf{IMM}_{\mathbf{w},d}$. The algorithm has four main steps:

1. *Computing irreducible invariant subspaces (Steps 2–6):* The algorithm starts by computing a basis of the Lie algebra $\mathfrak{g}_f$. It then invokes Algorithm 3 to compute bases of the d irreducible invariant subspaces of $\mathfrak{g}_f$. Algorithm 3 works by picking a random element R' in $\mathfrak{g}_f$ and factoring its characteristic polynomial $h = g_1 \cdots g_s$. By computing the closure of vectors (Definition 14) picked from null spaces of $g_1(R'), \ldots, g_s(R')$, the algorithm is able to find bases of the required invariant spaces.
2. *Computing layer spaces (Step 9)*: The direct relation between the irreducible invariant spaces of $\mathfrak{g}_{\mathsf{IMM}}$ and the layers spaces of any full rank ABP computing f (as shown in Lemma 49) is exploited by Algorithm 5 to compute bases of these layer spaces. This also helps establish that all the layer spaces, except two of them, are 'unique' (see Lemma 48). The second and second-to-last layer spaces of a full rank ABP are *not* unique; however the bigger space spanned by the first two layer spaces (similarly the last two layer spaces) is unique. Algorithm 5 finds bases for these two bigger spaces along with the $d-2$ remaining layer spaces.
3. *Reduction to almost set-multilinear ABP (Steps 12–15)*: The layer spaces are then correctly reordered in Algorithm 6 using a randomized procedure to compute the appropriate evaluation dimensions (Definition 16). The reordering also yields a valid width vector $\mathbf{w}$. At this point, the problem easily reduces to reconstructing a full rank almost set-multilinear ABP by mapping the bases of the layer spaces to distinct variables. This mapping gives an $\widehat{A} \in \mathsf{GL}(n)$ such that $f(\widehat{A}\mathbf{x})$ is computable by a full rank almost set-multilinear ABP of width $\mathbf{w}$. It is 'almost set-multilinear' (and not 'set-multilinear') as the second and the second-to-last layer spaces are unavailable; instead, two bigger spaces are available as mentioned above.
4. *Reconstructing a full rank almost set-mutlilinear ABP (Steps 18–22)*: Finally, we reconstruct a full rank almost set-mutlilinear ABP computing $f(\widehat{A}\mathbf{x})$ using Algorithm 7. This algorithm is inspired by a similar algorithm for reconstructing set-multilinear ABP in [29], but it is a little different from the latter as we are dealing with an 'almost' set-multilinear ABP. The reconstructed ABP readily gives an $A \in \mathsf{GL}(n)$ such that $f = \mathsf{IMM}_{\mathbf{w},d}(A\mathbf{x})$.

A final identity testing (Steps 25–30) takes care of the situation when, to begin with, there is no $\mathbf{w} \in \mathbb{N}^{d-1}$ that makes f equivalent to $\mathsf{IMM}_{\mathbf{w},d}$.

2 Preliminaries

2.1 Notations and definitions

The group of invertible $n \times n$ matrices over $\mathbb{F}$ is represented by $\mathsf{GL}(n, \mathbb{F})$. Since $\mathbb{F}$ is fixed to be the field of rationals, we omit $\mathbb{F}$ and write $\mathsf{GL}(n)$. Natural numbers are denoted by

Algorithm 2 Equivalence test for IMM

INPUT: Blackbox access to a homogeneous n variate degree d polynomial f (which can be assumed to be irreducible without any loss of generality).
OUTPUT: A $\mathbf{w} \in \mathbb{N}^{d-1}$ and an $A \in \mathsf{GL}(n)$ such that $f = \mathsf{IMM}_{\mathbf{w},d}(A\mathbf{x})$, if such $\mathbf{w}$ and A exist.

/* Finding irreducible invariant subspaces */
Compute a basis of the Lie algebra $\mathfrak{g}_f$. (See Section 2.2.)
Use Algorithm 3 to compute the bases of the irreducible invariant subspaces of $\mathfrak{g}_f$.
if Algorithm 3 outputs 'Fail' **then**
    Output 'no such $\mathbf{w}$ and A exist' and stop.
end if

/* Finding layer spaces from irreducible invariant subspaces */
Use Algorithm 5 to compute bases of the layer spaces of a full rank ABP computing f, if such an ABP exists.

/* Reduction to almost set-multilinear ABP: Finding $\mathbf{w}$ */
Use Algorithm 6 to compute a $\mathbf{w} \in \mathbb{N}^{d-1}$ and an $\widehat{A} \in \mathsf{GL}(n)$ such that $h = f(\widehat{A}\mathbf{x})$ is computable by a full rank almost set-multilinear ABP of width $\mathbf{w}$.
if Algorithm 6 outputs 'Fail' **then**
    Output 'no such $\mathbf{w}$ and A exist' and stop.
end if

/* Reconstructing an almost set-multilinear ABP: Finding A */
Use Algorithm 7 to reconstruct a full rank almost set-multilinear ABP $\mathtt{A}$' computing h.
if Algorithm 7 outputs 'Fail' **then**
    Output 'no such $\mathbf{w}$ and A exist' and stop.
end if
Replace the $\mathbf{x}$ variables in $\mathtt{A}$' by $\widehat{A}^{-1}\mathbf{x}$ to obtain a full rank ABP $\mathtt{A}$. Compute $A \in \mathsf{GL}(n)$ from $\mathtt{A}$.

/* Final identity testing */
Choose a point $\mathbf{a} \in S^n$, where $S \subseteq \mathbb{F}$ and $|S| \geq \text{poly}(n)$.
if $f(\mathbf{a}) \neq \mathsf{IMM}_{\mathbf{w},d}(A\mathbf{a})$ **then**
    Output 'no such $\mathbf{w}$ and A exist' and stop.
else
    Output $\mathbf{w}$ and A.
end if

$\mathbb{N} = \{1, 2, \dots\}$. As a convention, we use $\mathbf{x}, \mathbf{y}$ and $\mathbf{z}$ to denote sets of variables, capital letters A, B, C and so on to denote matrices, calligraphic letters like $\mathcal{U}, \mathcal{V}, \mathcal{W}$ to denote vector spaces over $\mathbb{F}$, and bold small letters like $\mathbf{u}, \mathbf{v}, \mathbf{w}$ to denote vectors in these spaces. All vectors considered in this article are column vectors, unless mentioned otherwise. An affine form in $\mathbf{x} = \{x_1, x_2, \dots, x_n\}$ variables is $a_0 + \sum_{i=1}^{n} a_i x_i$ where for $i \in [0, d]$ $a_i \in \mathbb{F}$, and if $a_0 = 0$ then we call it a *linear form*. The first order partial derivative of the polynomial $f(\mathbf{x})$ with respect to x_i is denoted as $\partial_{x_i}(f(\mathbf{x}))$. Below we set up some notations and terminologies.

2.1.1 Linear Algebra

▶ **Definition 9** (Direct sum)**.** Let $\mathcal{U}, \mathcal{W}$ be subspaces of a vector space $\mathcal{V}$. Then $\mathcal{V}$ is said to be the *direct sum* of $\mathcal{U}$ and $\mathcal{W}$ denoted $\mathcal{V} = \mathcal{U} \oplus \mathcal{W}$, if $V = \mathcal{U} + \mathcal{W}$ and $\mathcal{U} \cap \mathcal{W} = \{\mathbf{0}\}$.

For $\mathcal{U}, \mathcal{W}$ subspaces of a vector space $\mathcal{V}$, $\mathcal{V} = \mathcal{U} \oplus \mathcal{W}$ if and only if for every $\mathbf{v} \in \mathcal{V}$ there exist unique $\mathbf{u} \in \mathcal{U}$ and $\mathbf{w} \in \mathcal{W}$ such that $\mathbf{v} = \mathbf{u} + \mathbf{w}$. Hence, $\dim(\mathcal{V}) = \dim(\mathcal{U}) + \dim(\mathcal{W})$.

▶ **Definition 10** (Null space)**.** *Null space* $\mathcal{N}$ of a matrix $M \in \mathbb{F}^{n \times n}$ is the space of all vectors $\mathbf{v} \in \mathbb{F}^n$, such that $M\mathbf{v} = \mathbf{0}$.

▶ **Definition 11** (Coordinate subspace)**.** Let $e_i = (0, \ldots, 1, \ldots, 0)$ be the unit vector in $\mathbb{F}^n$ with 1 at the i-th position and all other coordinates zero. A *coordinate subspace* of $\mathbb{F}^n$ is a space spanned by a subset of the n unit vectors $\{e_1, e_2, \ldots, e_n\}$.

▶ **Definition 12** (Invariant subspace)**.** Let $M_1, M_2, \ldots, M_k \in \mathbb{F}^{n \times n}$. A subspace $\mathcal{U} \subseteq \mathbb{F}^n$ is called an *invariant subspace* of $\{M_1, M_2, \ldots, M_k\}$ if $M_i\,\mathcal{U} \subseteq \mathcal{U}$ for every $i \in [k]$. A nonzero invariant subspace $\mathcal{U}$ is *irreducible* if there are no invariant subspaces $\mathcal{U}_1$ and $\mathcal{U}_2$ such that $\mathcal{U} = \mathcal{U}_1 \oplus \mathcal{U}_2$, where $\mathcal{U}_1$ and $\mathcal{U}_2$ are properly contained in $\mathcal{U}$.

The following observation is immediate.

▶ **Observation 13.** *If $\mathcal{U}$ is an invariant subspace of $\{M_1, M_2, \ldots, M_k\}$ then for every $M \in \mathcal{L} \stackrel{\text{def}}{=} \text{span}_{\mathbb{F}}\{M_1, M_2, \ldots, M_k\}$, $M\,\mathcal{U} \subseteq \mathcal{U}$. Hence we say $\mathcal{U}$ is an invariant subspace of $\mathcal{L}$, a space generated by matrices.*

▶ **Definition 14** (Closure of a vector)**.** The *closure of a vector* $\mathbf{v} \in \mathbb{F}^n$ under the action of a space $\mathcal{L}$ spanned by a set of $n \times n$ matrices is the smallest invariant subspace of $\mathcal{L}$ containing $\mathbf{v}$.

Here, 'smallest' is with regard to dimension of invariant subspaces. Since intersection of two invariant subspaces is also an invariant subspace of $\mathcal{L}$, the smallest invariant subspace of $\mathcal{L}$ containing $\mathbf{v}$ is unique and is contained in every invariant subspace of $\mathcal{L}$ containing $\mathbf{v}$. Algorithm 4 in Section 4.2 computes the closure of a given vector $\mathbf{v}$ under the action of $\mathcal{L}$ whose basis is given.

By identifying a linear form $\sum_{i=1}^{n} a_i x_i$ with the vector $(a_1, \ldots, a_n) \in \mathbb{F}^n$ (and vice versa), we can associate the following vector spaces with an ABP.

▶ **Definition 15** (Layer spaces of an ABP)**.** Let $X_1 \cdot X_2 \ldots X_d$ be a full rank ABP $\mathtt{A}$ of length d and width $\mathbf{w} = (w_1, w_2, \ldots, w_{d-1})$, where X_1 to X_d are as in Definition 4. Let $\mathcal{X}_i$ be the vector space in $\mathbb{F}^n$ spanned by the homogeneous degree 1 parts of the affine forms[25] in X_i for $i \in [d]$; the spaces $\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_d$ are called the *layer spaces* of $\mathtt{A}$.

2.1.2 Evaluation dimension

The rank of the partial derivative matrix of a polynomial f was introduced in [34] and used subsequently in several works on lower bound, polynomial identity testing and circuit reconstruction (see [39]). The following definition (which makes the notion well defined for fields of finite characteristic) appears in [14][26].

[25] Identify linear forms with vectors in $\mathbb{F}^n$ as mentioned above.
[26] They attributed the definition to Ramprasad Saptharishi.

▶ **Definition 16** (Evaluation dimension). The *evaluation dimension* of a polynomial $g \in \mathbb{F}[\mathbf{x}]$ with respect to a set $\mathbf{x}' \subseteq \mathbf{x}$, denoted as $\text{Evaldim}_{\mathbf{x}'}(g)$, is defined as

$$\dim(\text{span}_{\mathbb{F}}\{g(\mathbf{x})|_{\forall x_j \in \mathbf{x}' \; x_j=\alpha_j} : \; \alpha_j \in \mathbb{F} \text{ for every } x_j \in \mathbf{x}'\}).$$

2.1.3 Affine projection and equivalence testing

Studying polynomials by applying linear transformations (from suitable matrix groups) on the variables is at the heart of invariant theory.

▶ **Definition 17** (Affine projection). An m variate polynomial f is an *affine projection* of a n variate polynomial g, if there exists a matrix $A \in \mathbb{F}^{n\times m}$ and a $\mathbf{b} \in \mathbb{F}^n$ such that $f(\mathbf{x}) = g(A\mathbf{x} + \mathbf{b})$.

In [25], it was shown that given an m variate polynomial f and an n variate polynomial g, checking whether f is an affine projection of g is NP-hard, even if f and g are given in the dense representation (that is as list of coefficients of the monomials). In the above definition, we say f is an affine projection of g via a *full rank transformation*, if $m \geq n$ and A has rank n. In the affine projection via full rank transformation problem, we are given an m variate polynomial f and an n variate polynomial g in some suitable representation, and we need to determine if f is an affine projection of g via a full rank transformation. [24, 25] studied the affine projection via full rank transformation problem for g coming from fixed families and gave polynomial time randomized algorithms to check whether a degree d polynomial f given as blackbox is an affine projection of g via a full rank transformation, where g is the elementary symmetric polynomial/permanent/determinant/power symmetric polynomial or sum-of-products polynomial. As observed in [25], variable reduction and translation equivalence test (described in Section 2.2) help reduce the affine projection via full rank transformation problem to equivalence testing (see also Section 1.5).

▶ **Definition 18** (Equivalent polynomials). An n variate polynomial f is *equivalent* to an n variate polynomial g, if there exists a matrix $A \in \mathsf{GL}(n)$ such that $f(\mathbf{x}) = g(A\mathbf{x})$.

The equivalence testing problem asks us to check if two n variate polynomials f and g (given in some suitable representation) are equivalent. This problem is at least as hard as the graph isomorphism problem even when f and g are cubic forms given in dense representation [4]. There is a cryptographic application [35] that assumes the problem is hard also in the *average-case* for bounded degree f and g given in dense representation. If we restrict to checking if f and g are equivalent via a permutation matrix A, then the problem is shown to be in $\mathsf{NP} \cap \mathsf{coAM}$ [41].

2.1.4 Group of symmetries and Lie algebra

▶ **Definition 19** (Group of symmetries). The *group of symmetries* of a polynomial $g \in \mathbb{F}[\mathbf{x}]$ in n variables, denoted as $\mathcal{G}_g$, is the set of all $A \in \mathsf{GL}(n)$ such that $g(A\mathbf{x}) = g(\mathbf{x})$.

The proof of Theorem 7 involves an analysis of the Lie algebra of the group of symmetries of $\mathsf{IMM}_{\mathbf{w},d}$. We will abuse terminology slightly and say the Lie algebra of a polynomial to mean the Lie algebra of the group of symmetries of the polynomial. We will work with the following definition of Lie algebra of a polynomial (see [25]).

▶ **Definition 20** (Lie algebra of a polynomial). The *Lie algebra* of a polynomial $f \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ denoted as $\mathfrak{g}_f$ is the set of all $n \times n$ matrices $E = (e_{ij})_{i,j\in[n]}$ in $\mathbb{F}^{n\times n}$ such that $\sum_{i,j\in[n]} e_{ij}x_j \cdot \frac{\partial f}{\partial x_i} = 0$.

Remark: Observe that $\mathfrak{g}_f$ is a subspace of $\mathbb{F}^{n\times n}$. It can also be shown that the space $\mathfrak{g}_f$ satisfies the *Lie bracket property*: For any $E_1, E_2 \in \mathfrak{g}_f$, $[E_1, E_2] \stackrel{\text{def}}{=} E_1E_2 - E_2E_1$ is also in $\mathfrak{g}_f$. We would not be needing this property, but would just use the vector space feature of $\mathfrak{g}_f$. The proof of the following well known fact is given in [25], see also Section 7.1 for a proof.

▶ **Claim 21.** *If $f(\mathbf{x}) = g(A\mathbf{x})$, where f and g are both n variate polynomials and $A \in \mathsf{GL}(n)$, then the Lie algebra of f is a conjugate of the Lie algebra of g via A, i.e. $\mathfrak{g}_f = \{A^{-1}EA \,:\, E \in \mathfrak{g}_g\} =: A^{-1}\mathfrak{g}_g A$.*

The following observation relates the invariant subspaces of the Lie algebras of two equivalent polynomials.

▶ **Observation 22.** *Suppose $f(\mathbf{x}) = g(A\mathbf{x})$, where $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$ and $A \in \mathsf{GL}(n)$. Then $\mathcal{U} \in \mathbb{F}^n$ is an invariant subspace of $\mathfrak{g}_g$ if and only if $A^{-1}\mathcal{U}$ is an invariant subspace of $\mathfrak{g}_f$.*

Proof. $\mathcal{U}$ is an invariant subspace of $\mathfrak{g}_g$ implies, for all $E \in \mathfrak{g}_g$, $E\,\mathcal{U} \subseteq \mathcal{U}$. Consider $E' \in \mathfrak{g}_f$, using Claim 21 we know there exists $E \in \mathfrak{g}_g$ such that $AE'A^{-1} = E$. Since $\mathcal{U}$ is an invariant subspace of $AE'A^{-1}$, $A^{-1}\mathcal{U}$ is an invariant subspace of E'. The proof of the other direction is similar. ◀

2.2 Algorithmic preliminaries

We record some of the basic algorithmic tasks on polynomials that can be performed efficiently and which we require at different places in our algorithms and proofs.

2.2.1 Computing homogeneous components of f

The i-th homogeneous component (or the homogeneous degree i part) of a degree d polynomial f, denoted as $f^{[i]}$ is the sum of the degree i monomials with coefficients as in f. Clearly, $f = f^{[d]} + f^{[d-1]} + \cdots + f^{[0]}$. Given an n variate degree d polynomial f as a blackbox, there is an efficient algorithm to compute blackboxes for the d homogeneous components of f. The idea is to multiply each variable by a new formal variable t, and then interpolate the coefficients of $t^0, t^1, \ldots, t^d$; the coefficient of t^i is $f^{[i]}$.

2.2.2 Computing derivatives of f

Given a polynomial $f(x_1, x_2, \ldots, x_n)$ of degree d as a blackbox, we can efficiently construct blackboxes for the derivatives $\partial_{x_i} f$, for all $i \in [n]$. The following observation suggests that it is sufficient to construct blackboxes for certain homogeneous components.

▶ **Observation 23.** *If $g(x_1, x_2, \ldots, x_n)$ is a homogeneous polynomial of degree d then for all $i \in [n]$ $\partial_{x_i} g = \sum_{j=1}^{d} j \cdot x_i^{j-1}[g(x_1, x_2, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_n)]^{[d-j]}$.*

For every $i \in [n]$, constructing a blackbox for $\partial_{x_i} f$ is immediate from the above observation as $\partial_{x_i} f = \partial_{x_i} f^{[d]} + \partial_{x_i} f^{[d-1]} + \cdots + \partial_{x_i} f^{[1]}$.

2.2.3 Space of linear dependencies of polynomials

Let $f_1, f_2, \ldots, f_m$ be n variate polynomials in $\mathbb{F}[\mathbf{x}]$ with degree bounded by d. The set $\mathcal{U} = \{(a_1\ a_2 \ldots\ a_m)^T \in \mathbb{F}^m \mid \sum_{j\in[m]} a_j f_j = 0\}$, called the space of $\mathbb{F}$-linear dependencies of $f_1, f_2, \ldots, f_m$ is a subspace of $\mathbb{F}^m$. We would like to find a basis of the space $\mathcal{U}$ given

blackbox access to $f_1, f_2, \ldots, f_m$. Suppose the dimension of the $\mathbb{F}$-linear space spanned by the polynomials $f_1, f_2, \ldots, f_m$ is $m - r$ then $\dim(\mathcal{U}) = r$. An algorithm to find a basis of $\mathcal{U}$ can be derived from the following claim.

▶ **Claim 24.** *With probability at least* $1 - \frac{1}{\text{poly}(n)}$*, the rank of the matrix* $M = (f_j(\mathbf{b}_i))_{i,j\in[m]}$ *is* $m - r$ *where* $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_m$ *are chosen independently and uniformly at random from* $S^n \subset \mathbb{F}^n$ *with* $|S| = dm \cdot \text{poly}(n)$*.*

The proof of the claim which involves an application of the Schwartz-Zippel lemma is given in Section 7.1. The space $\mathcal{U}$ equals the null space of M with high probability.

2.2.4 Eliminating redundant variables

▶ **Definition 25** (Essential and redundant variables)**.** We say an n variate polynomial f has s *essential variables* if there exists an $A \in \mathsf{GL}(n)$ such that $f(A\mathbf{x})$ is an s variate polynomial and there exists no $A' \in \mathsf{GL}(n)$ such that $f(A'\mathbf{x})$ is a t variate polynomial where $t < s$. An n variate polynomial has r *redundant variables* if it has $s = n - r$ essential variables.

If the number of essential variables in a polynomial $f(x_1, x_2, \ldots, x_n)$ is s then without loss of generality we can assume that the first s variables $x_1, x_2, \ldots, x_s$ are essential variables and the remaining variables are redundant. An algorithm to eliminate the redundant variables of a polynomial was considered in [12], and it was shown that if the coefficients of a polynomial are given as input then we can eliminate the redundant variables in polynomial time. Further, [24] gave an efficient randomized algorithm to eliminate the redundant variables in a polynomial given as blackbox. For completeness, we give the algorithm in [24] as part of the following claim.

▶ **Claim 26.** *Let* r *be the number of redundant variables in an* n *variate polynomial* f *of degree* d*. Then the dimension of the space* $\mathcal{U}$ *of* $\mathbb{F}$*-linear dependencies of* $\{\partial_{x_i} f \mid i \in [n]\}$ *is* r*. Moreover, we can construct an* $A \in \mathsf{GL}(n)$ *in randomized* $\text{poly}(n, d, \beta)$ *time such that* $f(A\mathbf{x})$ *is free of the set of variables* $\{x_{n-r+1}, x_{n-r+2}, \ldots, x_n\}$*, where* β *is the bit length of the coefficients of* f*.*

The proof is given in Section 7.1.

2.2.5 Efficient translation equivalence test

Two n variate degree d polynomials $f, g \in \mathbb{F}[\mathbf{x}]$ are *translation equivalent* (also called shift equivalent in [13]) if there exists a point $\mathbf{a} \in \mathbb{F}^n$ such that $f(\mathbf{x} + \mathbf{a}) = g(\mathbf{x})$. Translation equivalence test takes input blackbox access to two n variate polynomials f and g, and outputs an $\mathbf{a} \in \mathbb{F}^n$ such that $f(\mathbf{x} + \mathbf{a}) = g(\mathbf{x})$ if f and g are translation equivalent else outputs 'f and g are not translation equivalent'. As before, let β be the bit lengths of the coefficients of f and g. A randomized $\text{poly}(n, d, \beta)$ time algorithm is presented in [13] to test translation equivalence and find an $\mathbf{a} \in \mathbb{F}^n$ such that $f(\mathbf{x} + \mathbf{a}) = g(\mathbf{x})$, if such an $\mathbf{a}$ exists. Another randomized test was mentioned in [25], which we present as proof of the following lemma in Section 7.1.

▶ **Lemma 27.** *There is a randomized algorithm that takes input blackbox access to two* n *variate, degree* d *polynomials* f *and* g*, and with probability at least* $1 - \frac{1}{\text{poly}(n)}$ *does the following: if* f *is translation equivalent to* g*, outputs an* $\mathbf{a} \in \mathbb{F}^n$ *such that* $f(\mathbf{x} + \mathbf{a}) = g(\mathbf{x})$*, else outputs '*$f$ *and* g *are not translation equivalent'. The running time of the algorithm is* $\text{poly}(n, d, \beta)$*, where* β *is the bit length of the coefficients of* f *and* g*.*

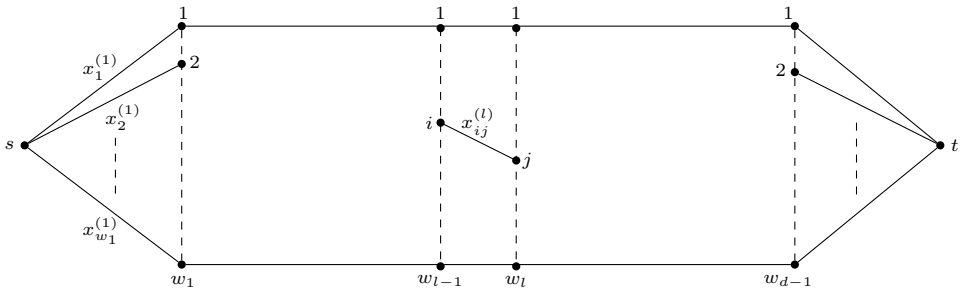


Figure 1 Naming of variables in $\mathsf{IMM}_{\mathbf{w},d}$.

2.2.6 Computing basis of Lie algebra

The proof of the following lemma is given in [25], for completeness we include a proof in Section 7.1.

▶ **Lemma 28.** *There is a randomized algorithm which when given blackbox access to an n variate degree d polynomial f, computes a basis of $\mathfrak{g}_f$ with probability at least $1 - \frac{1}{\text{poly}(n)}$ in time $\text{poly}(n, d, \beta)$ where β is the bit length of the coefficients in f.*

2.3 Iterated matrix multiplication polynomial

Let $\mathbf{w} = (w_1, w_2, \ldots, w_{d-1}) \subseteq \mathbb{N}^{d-1}$. Suppose $Q_1 = (x_1^{(1)}\ x_2^{(1)} \ldots\ x_{w_1}^{(1)})$, $Q_d^T = (x_1^{(d)}\ x_2^{(d)} \ldots\ x_{w_{d-1}}^{(d)})$ be row vectors, and for $k \in [2, d-1]$, $Q_k = (x_{ij}^{(k)})_{i\in[w_{k-1}],j\in[w_k]}$ be a $w_{k-1} \times w_k$ matrix, where for $i \in [w_1]$ $x_i^{(1)}$, for $i \in [w_{d-1}]$ $x_i^{(d)}$ and for $i \in [w_{k-1}], j \in [w_k]$ $x_{ij}^{(k)}$ are distinct variables. The iterated matrix multiplication polynomial $\mathsf{IMM}_{\mathbf{w},d}$ is the entry of the 1×1 matrix obtained from the product $\prod_{i=1}^{d} Q_i$. When d and $\mathbf{w}$ are clear from the context, we drop the subscripts and simply represent it by IMM. For all $k \in [d]$, we denote the set of variables in Q_k as $\mathbf{x}_k$; Figure 1 depicts an ABP computing $\mathsf{IMM}_{\mathbf{w},d}$ when the width is uniform, that is $w_1 = w_2 = \cdots = w_{d-1}$.

Ordering of variables in $\mathsf{IMM}_{\mathbf{w},d}$: From here on we will assume that the variables $\mathbf{x}_1 \uplus \mathbf{x}_2 \uplus \cdots \uplus \mathbf{x}_d$ are ordered as follows: For $i < j$, the $\mathbf{x}_i$ variables have precedence over the $\mathbf{x}_j$ variables. Among the $\mathbf{x}_l$ variables, we follow column-major ordering, i.e $x_{11}^{(l)} \succ \cdots \succ x_{w_{l-1}1}^{(l)} \succ \cdots \succ x_{1w_l}^{(l)} \succ \cdots \succ x_{w_{l-1}w_l}^{(l)}$. We would also refer to the variables of IMM as $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$ where x_i is the i-th variable according to this ordering[27], and $n = w_1 + \sum_{k=2}^{d-1} w_{k-1}w_k + w_{d-1}$ is the total number of variables in IMM. For $A \in \mathbb{F}^{n\times n}$ we can naturally index the rows and columns of A by the $\mathbf{x}$ variables such that the i-th row or column is indexed by the i-th variable.

2.4 Almost set-multilinear ABP and a canonical representation

In the proof of Theorem 7, we eventually reduce the equivalence test problem to checking whether there exists an $A \in \mathsf{GL}(n)$, such that an input polynomial $h(\mathbf{x})$ (given as blackbox)

[27] The justification for identifying the variables $\mathbf{x}$ of f with the variables of $\mathsf{IMM}_{\mathbf{w},d}$ in this order is as follows: If f is equivalent to $\mathsf{IMM}_{\mathbf{w},d}$ then f is also equivalent to $\mathsf{IMM}_{\mathbf{w},d}(\mathbf{x})$ whose variables $\{x_1, \ldots, x_n\}$ are ordered as above. That $\mathbf{w}$ is a priori unknown to Algorithm 2 does not matter here.

equals $\mathsf{IMM}_{\mathbf{w},d}(A\mathbf{x})$, where $\mathbf{w}$ is known, $\mathbf{x}$ is the variables of $\mathsf{IMM}_{\mathbf{w},d}$, and A satisfies the following properties:

1. For all $k \in [d] \setminus \{2, d-1\}$, the rows indexed by $\mathbf{x}_k$ variables contain zero entries in columns indexed by variables other than $\mathbf{x}_i$.
2. The rows indexed by $\mathbf{x}_2$ and $\mathbf{x}_{d-1}$ variables contain zero entries in columns indexed by variables other than $\mathbf{x}_1 \uplus \mathbf{x}_2$ and $\mathbf{x}_{d-1} \uplus \mathbf{x}_d$ respectively.

If there exists such a block-diagonal matrix A then we say h is computed by a *full rank almost set-multilinear* ABP as defined below.

▶ **Definition 29** (Full rank almost set-multilinear ABP)**.** A *full rank almost set-multilinear* ABP of width $\mathbf{w} = (w_1, w_2, \ldots, w_{d-1})$ and length d is a product of d matrices, $X_1 \cdot X_2 \ldots X_d$, where X_k's are as in Definition 4 but with linear forms as entries. The linear forms in X_k are in $\mathbf{x}_k$ variables, for all $k \in [d] \setminus \{2, d-1\}$, and for X_2 and X_{d-1} the linear forms are in $\mathbf{x}_1 \uplus \mathbf{x}_2$ and $\mathbf{x}_{d-1} \uplus \mathbf{x}_d$ variables respectively, where $\mathbf{x}_1 \uplus \mathbf{x}_2 \cdots \uplus \mathbf{x}_d$ is the set of variables in $\mathsf{IMM}_{\mathbf{w},d}$.

Conventionally, in the definition of set-multilinear ABP, the entries of X_i are linear forms in just $\mathbf{x}_i$ variables – the ABP in the above definition is almost set-multilinear as matrices X_2 and X_{d-1} violate this condition. An efficient randomized reconstruction algorithm for set-multilinear ABP follows from [29]. In order to apply a similar reconstruction algorithm to full rank almost set-multilinear ABPs, we fix a canonical representation for the first two and the last two matrices as explained below.

Canonical form or representation: We say a full rank almost set-multilinear ABP of width $\mathbf{w}$ is in *canonical form* if the following hold:

(1a) $X_1 = (x_1^{(1)}\ x_2^{(1)} \ldots\ x_{w_1}^{(1)})$,

(1b) the linear forms in X_2 are such that for $l, i \in [w_1]$ and $l < i$, the variable $x_l^{(1)}$ has a zero coefficient in the (i,j)-th entry (linear form) of X_2, where $j \in [w_2]$.

(2a) $X_d = (x_1^{(d)}\ x_2^{(d)} \ldots\ x_{w_{d-1}}^{(d)})^T$,

(2b) the linear forms in X_{d-1} are such that for $l, j \in [w_{d-1}]$ and $l < j$, the variable $x_l^{(d)}$ has a zero coefficient in the (i,j)-th entry (linear form) of X_{d-1}, where $i \in [w_{d-2}]$.

The following claim states that for every full rank almost set-multilinear ABP there is another ABP in canonical form computing the same polynomial, and the latter can be computed efficiently.

▶ **Claim 30.** *Let h be an n variate, degree d polynomial computable by a full rank almost set-multilinear ABP of width $\mathbf{w} = (w_1, w_2, \ldots, w_{d-1})$ and length d. There is a randomized algorithm that takes input blackbox access to h and the width vector $\mathbf{w}$, and outputs a full rank almost set-multilinear ABP of width $\mathbf{w}$ in canonical form computing h, with probability at least $1 - \frac{1}{\mathrm{poly}(n)}$. The running time of the algorithm is $\mathrm{poly}(n, \beta)$, where β is the bit length of the coefficients of h.*

We prove the claim in Section 5.3. The algorithm is similar to reconstruction of set-multilinear ABP in [29], except that the latter needs to be adapted suitably as we are dealing with almost set-multilinear ABP.

3 Lie algebra of IMM

Dropping the subscripts $\mathbf{w}$ and d, we refer to $\mathsf{IMM}_{\mathbf{w},d}$ as IMM. We show that the Lie algebra, $\mathfrak{g}_{\mathsf{IMM}}$ consists of well-structured subspaces and by analysing these subspaces we are able to identify all the irreducible invariant subspaces of $\mathfrak{g}_{\mathsf{IMM}}$.

3.1 Structure of the Lie algebra $\mathfrak{g}_{\mathsf{IMM}}$

Recall that $\mathbf{x} = \mathbf{x}_1 \uplus \mathbf{x}_2 \uplus \cdots \uplus \mathbf{x}_d$ are the variables of IMM which are also referred to as $\{x_1, x_2, \ldots, x_n\}$[28] for notational convenience.

▶ **Lemma 31.** *Let $\mathcal{W}_1, \mathcal{W}_2, \mathcal{W}_3$ be the following sets (spaces) of matrices:*

1. *$\mathcal{W}_1$ consists of all matrices $D = (d_{ij})_{i,j\in[n]}$ such that D is diagonal and*

$$\sum_{i=1}^{n} d_{ii} x_i \cdot \frac{\partial \mathsf{IMM}}{\partial x_i} = 0.$$

2. *$\mathcal{W}_2$ consists of all matrices $B = (b_{ij})_{i,j\in[n]}$ such that*

$$\sum_{i,j\in[n]} b_{ij} x_j \cdot \frac{\partial \mathsf{IMM}}{\partial x_i} = 0,$$

   where in every summand $b_{ij} \neq 0$ only if $x_i \neq x_j$ and $x_i, x_j \in \mathbf{x}_l$ for some $l \in [d]$.

3. *$\mathcal{W}_3$ consists of all matrices $C = (c_{ij})_{i,j\in[n]}$ such that*

$$\sum_{i,j\in[n]} c_{ij} x_j \cdot \frac{\partial \mathsf{IMM}}{\partial x_i} = 0,$$

   where in every summand $c_{ij} \neq 0$ only if either $x_i \in \mathbf{x}_2$, $x_j \in \mathbf{x}_1$ or $x_i \in \mathbf{x}_{d-1}$, $x_j \in \mathbf{x}_d$.

Then $\mathfrak{g}_{\mathsf{IMM}} = \mathcal{W}_1 \oplus \mathcal{W}_2 \oplus \mathcal{W}_3$.

The proof of Lemma 31 is given in Section 7.2.

Elaboration on Lemma 31: An element $E = (e_{ij})_{i,j\in[n]}$ of $\mathfrak{g}_{\mathsf{IMM}}$ is an $n \times n$ matrix with rows and columns indexed by variables of IMM following the ordering mentioned in Section 2.3. Since $\sum_{i,j\in[n]} e_{ij} x_j \cdot \frac{\partial \mathsf{IMM}}{\partial x_i} = 0$, E appears as shown in Figure 2, where the row indices correspond to derivatives and column indices correspond to *shifts*[29].

The proof will show that E is a sum of three matrices $D \in \mathcal{W}_1$, $B \in \mathcal{W}_2$ and $C \in \mathcal{W}_3$ such that

1. D contributes to the diagonal entries.
2. B contributes to the block-diagonal entries of E corresponding to the locations:
   - $(x_i^{(1)}, x_j^{(1)})$ where $i, j \in [w_1]$ and $i \neq j$
   - $(x_i^{(d)}, x_j^{(d)})$ where $i, j \in [w_{d-1}]$ and $i \neq j$
   - $(x_{ij}^{(l)}, x_{pq}^{(l)})$ where $i, p \in [w_{l-1}]$ and $j, q \in [w_l]$ for $l \in [2, d-1]$, and $(i, j) \neq (p, q)$.
3. C contributes to the two corner rectangular blocks corresponding to:
   - rows labelled by $\mathbf{x}_2$ variables and columns labelled by $\mathbf{x}_1$ variables
   - rows labelled by $\mathbf{x}_{d-1}$ variables and columns labelled by $\mathbf{x}_d$ variables.

In order to get a finer understanding of $\mathfrak{g}_{\mathsf{IMM}}$ and its dimension we look at the spaces $\mathcal{W}_1, \mathcal{W}_2$ and $\mathcal{W}_3$ closely, and henceforth call them the *diagonal space*, the *block-diagonal space* and the *corner space* respectively.

[28] Following the ordering mentioned in Section 2.3.
[29] Borrowing terminology from the *shifted partial derivatives* measure [26].

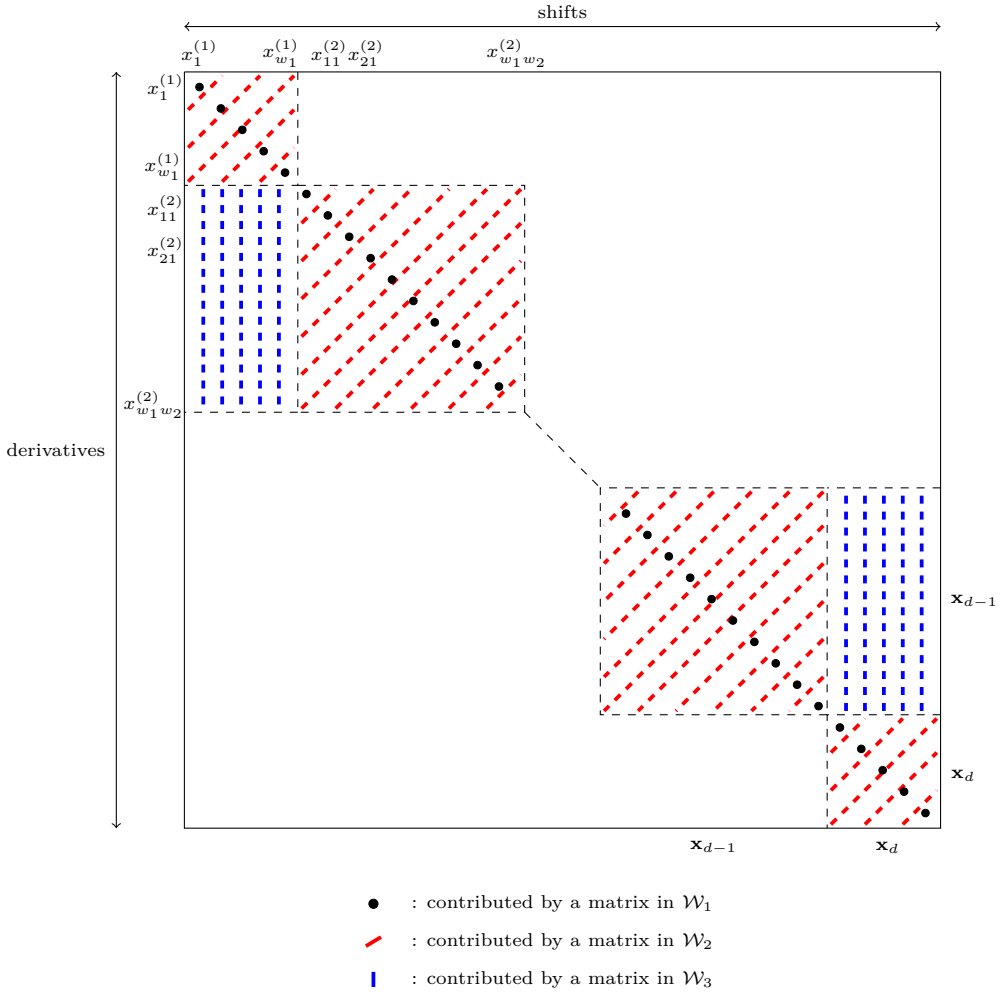


Figure 2 A matrix E in $\mathfrak{g}_{\mathsf{IMM}}$.

Corner space $\mathcal{W}_3$

▶ **Lemma 32** (Corner space). *The space* $\mathcal{W}_3 = \mathcal{W}_3^{(a)} \oplus \mathcal{W}_3^{(b)}$ *where* $\mathcal{W}_3^{(a)} = \mathcal{A}_1 \oplus \mathcal{A}_2 \oplus \cdots \oplus \mathcal{A}_{w_2}$ *and* $\mathcal{W}_3^{(b)} = \mathcal{A'}_1 \oplus \mathcal{A'}_2 \oplus \cdots \oplus \mathcal{A'}_{w_{d-2}}$ *such that for every* $i \in [w_2]$ $\mathcal{A}_i$ *is isomorphic to the space of* $w_1 \times w_1$ *anti-symmetric matrices over* $\mathbb{F}$*, and for every* $j \in [w_{d-2}]$ $\mathcal{A'}_j$ *is isomorphic to the space of* $w_{d-1} \times w_{d-1}$ *anti-symmetric matrices over* $\mathbb{F}$*. Hence* $\dim(\mathcal{W}_3) = \frac{1}{2}\left[w_1w_2(w_1-1) + w_{d-1}w_{d-2}(w_{d-1}-1)\right]$.

The proof is in Section 7.2. We briefly elaborate on the statement here.

Elaboration on Lemma 31: Every element $C \in \mathcal{W}_3$ can be expressed as a sum of two $n \times n$ matrices $C^{(a)} \in \mathcal{W}_3^{(a)}$ and $C^{(b)} \in \mathcal{W}_3^{(b)}$. $C^{(a)}$ looks as shown in Figure 3, where for every $i \in [w_2]$ $C_i^{(a)}$ is an anti-symmetric matrix. The structure of $C^{(b)}$ is similar[30] to that of $C^{(a)}$

[30] Once we rearrange the rows in $C^{(b)}$ indexed by variables in $\mathbf{x}_{d-1}$ according to row major ordering (instead of column major ordering) of variables in $\mathbf{x}_{d-1}$.

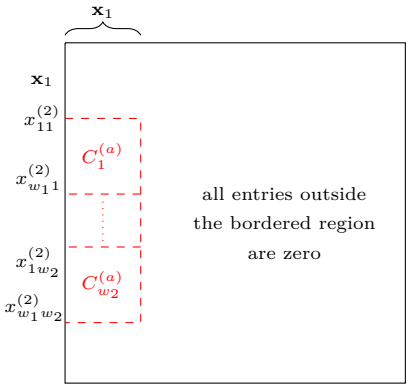


Figure 3 A matrix $C^{(a)}$ in $\mathcal{W}_3^{(a)}$.

with non zero entries restricted to the rows indexed by $\mathbf{x}_{d-1}$ variables and columns indexed by $\mathbf{x}_d$ variables.

Block-diagonal space $\mathcal{W}_2$

In the following lemma, $\mathcal{Z}_{w_k}$ denotes the space of $w_k \times w_k$ matrices with diagonal entries zero for $k \in [d-1]$. Also, for notational convenience we assume that $w_0 = w_d = 1$. We will also use the tensor product of matrices: if $A = (a_{i,j}) \in \mathbb{F}^{r \times s}$ and $B \in \mathbb{F}^{t \times u}$, then $A \otimes B$ is the $(rt) \times (su)$ matrix given by

$$A \otimes B = \begin{bmatrix} a_{1,1}B & \cdots & a_{1,s}B \\ \vdots & \vdots & \vdots \\ a_{r,1}B & \cdots & a_{r,s}B \end{bmatrix}.$$

▶ **Lemma 33** (Block-diagonal space). *The space $\mathcal{W}_2 = \mathcal{B}_1 \oplus \mathcal{B}_2 \oplus \cdots \oplus \mathcal{B}_{d-1}$ such that for every $k \in [d-1]$, $\mathcal{B}_k$ is isomorphic to the $\mathbb{F}$-linear space spanned by $t_k \times t_k$ matrices of the form*

$$\begin{bmatrix} -Z^T \otimes I_{w_{k-1}} & 0 \\ 0 & I_{w_{k+1}} \otimes Z \end{bmatrix}_{t_k \times t_k} \quad \text{where } Z \in \mathcal{Z}_{w_k} \text{ and } t_k = w_k(w_{k-1} + w_{k+1}). \tag{1}$$

Hence, $\dim(\mathcal{W}_2) = \sum_{k=1}^{d-1}(w_k^2 - w_k)$.

The proof is in Section 7.2.

Elaboration on Lemma 33: An element $B \in \mathcal{W}_2$ is a sum of $d-1$, $n \times n$ matrices $B_1, B_2, \ldots, B_{d-1}$ such that for every $k \in [d-1]$, $B_k \in \mathcal{B}_k$ and the non zero entries of B_k are restricted to the rows and columns indexed by $\mathbf{x}_k \uplus \mathbf{x}_{k+1}$ variables. The submatrix in B_k corresponding to these rows and columns looks as shown in Equation (1).

Diagonal space $\mathcal{W}_1$

In the next lemma, $\mathcal{Y}_{w_k}$ denotes the space of $w_k \times w_k$ diagonal matrices for $k \in [d-1]$. As before we assume $w_0 = w_d = 1$.

▶ **Lemma 34** (Diagonal Space). *The space $\mathcal{W}_1$ contains the space $\mathcal{D}_1 \oplus \mathcal{D}_2 \oplus \cdots \oplus \mathcal{D}_{d-1}$ such that for every $k \in [d-1]$, $\mathcal{D}_k$ is isomorphic to the $\mathbb{F}$-linear space spanned by $t_k \times t_k$ matrices of the form*

$$\begin{bmatrix} -Y \otimes I_{w_{k-1}} & 0 \\ 0 & I_{w_{k+1}} \otimes Y \end{bmatrix}_{t_k \times t_k} \quad \text{where } Y \in \mathcal{Y}_{w_k} \text{ and } t_k = w_k(w_{k-1} + w_{k+1}). \tag{2}$$

Hence, $\dim(\mathcal{W}_1) \geq \sum_{k=1}^{d-1} w_k$.

The proof (still given in Section 7.2) is similar to that of Lemma 33.

Elaboration on Lemma 34: An element $D \in \mathcal{D}_1 \oplus \mathcal{D}_2 \oplus \cdots \oplus \mathcal{D}_{d-1}$ is a sum of $d-1$, $n \times n$ matrices $D_1, D_2, \ldots, D_{d-1}$ such that for every $k \in [d-1]$, $D_k \in \mathcal{D}_k$ and the non zero entries of D_k are restricted to the rows and columns indexed by $\mathbf{x}_k \uplus \mathbf{x}_{k+1}$ variables. The submatrix in D_k corresponding to these rows and columns looks as shown in Equation (2).

3.2 Random elements of $\mathfrak{g}_{\mathsf{IMM}}$

The algorithm in Theorem 7 involves picking a random matrix R' in $\mathfrak{g}_f$ and computing its characteristic polynomial $h(x)$. To ensure the correctness of the algorithm, $h(x)$ will have to be square free over $\mathbb{F}$. In Lemma 36 we show that the characteristic polynomial of a random matrix R in $\mathfrak{g}_{\mathsf{IMM}}$ is square free with high probability. From Claim 21 this implies that if f is equivalent to IMM then the characteristic polynomial of R' is also square free with high probability.

▶ **Claim 35.** *There is a diagonal matrix $D \in \mathfrak{g}_{\mathsf{IMM}}$ with all entries distinct.*

Proof. From Lemma 34, we know that for $k \in [d-1]$ the submatrix of $D_k \in \mathcal{D}_k$ defined by the rows and columns indexed by the variables in $\mathbf{x}_k \uplus \mathbf{x}_{k+1}$ is

$$\begin{bmatrix} -Y_k \otimes I_{w_{k-1}} & 0 \\ 0 & I_{w_{k+1}} \otimes Y_k \end{bmatrix},$$

where $Y_k \in \mathcal{Y}_k$. Let the (i,i)-th entry of Y_k be $y_i^{(k)}$ and pretend that these entries are distinct formal variables, say $\mathbf{y}$ variables. Consider the matrix $D = \sum_{i=1}^{d-1} D_i$ and observe the following:

(a) For $k \in [2, d-1]$, the $(x_{ij}^{(k)}, x_{ij}^{(k)})$-th entry of D is $y_i^{(k-1)} - y_j^{(k)}$ where $i \in [w_{k-1}]$ and $j \in [w_k]$.

(b) The $(x_i^{(1)}, x_i^{(1)})$-th and $(x_j^{(d)}, x_j^{(d)})$-th entry of D are $-y_i^{(1)}$ and $y_j^{(d-1)}$ respectively, where $i \in [w_1]$ and $j \in [w_{d-1}]$.

In particular, all the diagonal entries of D are distinct linear forms in the $\mathbf{y}$ variables. Hence, if we assign values to the $\mathbf{y}$ variables uniformly at random from a set $S \subseteq \mathbb{F}$ such that $|S| \geq n^2$ then with non zero probability D has all diagonal entries distinct after the random assignment. ◀

▶ **Lemma 36.** *If $\{L_1, L_2, \ldots, L_m\}$ is a basis of the Lie algebra $\mathfrak{g}_{\mathsf{IMM}}$ then the characteristic polynomial of an element $L = \sum_{i=1}^{m} r_i L_i$, where $r_i \in_R \mathbb{F}$ is picked independently and uniformly at random from $[2n^3]$, is square free with probability at least $1 - \frac{1}{\mathrm{poly}(n)}$.*

Proof. Pretend that the r_i's are formal variables. The characteristic polynomial $h_{\mathbf{r}}(x)$ of L is a polynomial in x with coefficients that are polynomial of degree at most n in $\mathbf{r} = \{r_1, r_2, \ldots, r_m\}$ variables.

▶ **Observation 37.** *The discriminant of* $h_{\mathbf{r}}(x)$, $\mathrm{disc}(h_{\mathbf{r}}(x)) \coloneqq \mathrm{res}_x(h_{\mathbf{r}}, \frac{\partial h_{\mathbf{r}}}{\partial x})$, *is a non zero polynomial in* $\mathbf{r}$ *variables of degree at most* n^2, *where* $\mathrm{res}_x(h_{\mathbf{r}}, \frac{\partial h_{\mathbf{r}}}{\partial x})$ *is the resultant of* $h_{\mathbf{r}}$ *and* $\frac{\partial h_{\mathbf{r}}}{\partial x}$ *when treated as univariates in* x.

Proof. $h_{\mathbf{r}} = \mathsf{Det}(xI_n - r_1L_1 - \ldots - r_\omega L_\omega)$ is a degree n homogeneous polynomial in the variables $x, r_1, \ldots, r_\omega$. Let $S \in \mathbb{F}[\mathbf{r}]^{(2n-1)^2}$ be the Sylvester matrix of $h_{\mathbf{r}}$ and $\frac{\partial h_{\mathbf{r}}}{\partial x}$ with respect to x, i.e.

$$S_{i,j} = \begin{cases} [x^{n+i-j}]h_{\mathbf{r}} & \text{if } 1 \le i \le n-1 \\ [x^{i-j}]\frac{\partial h_{\mathbf{r}}}{\partial x} & \text{otherwise} \end{cases}$$

where $[x^\delta]g$ is the coefficient of the monomial x^δ in the polynomial g. Moreover, by homogeneity of $h_{\mathbf{r}}$, $[x^\delta]h_{\mathbf{r}}$ (resp. $[x^\delta]\frac{\partial h_{\mathbf{r}}}{\partial x}$) is a homogeneous polynomial of degree $(n-\delta)$ (resp. $(n-1-\delta)$) with respect to the variables $\mathbf{r}$. Then, if σ is a permutation of $[2n-1]$ then $\prod_{i=1}^{2n-1} S_{i,\sigma(i)}$ is a homogeneous polynomial in $\mathbf{r}$ of degree

$$\sum_{i=1}^{n-1} -i + \sigma(i) + \sum_{i=n}^{2n-1} n - 1 - i + \sigma(i) = n(n-1) - \left(\sum_{i=1}^{2n-1} i\right) + \left(\sum_{i=1}^{2n-1} \sigma(i)\right) = n(n-1).$$

Consequently, $\mathrm{disc}(h_{\mathbf{r}}(x))$ is homogeneous and of degree $n(n-1)$. If $\mathrm{res}_x(h_{\mathbf{r}}, \frac{\partial h_{\mathbf{r}}}{\partial x})$ is identically zero as a polynomial in $\mathbf{r}$ then for every setting of $\mathbf{r}$ to field elements $\gcd(h_{\mathbf{r}}, \frac{\partial h_{\mathbf{r}}}{\partial x}) \neq 1$ implying $h_{\mathbf{r}}$ is not square free. This would contradict Claim 35 as we can set the $\mathbf{r}$ variables appropriately such that L is a diagonal matrix with distinct diagonal entries, and $h_{\mathbf{r}}$ for such a setting of the $\mathbf{r}$ variables is square free. ◀

Since $\mathrm{disc}(h_{\mathbf{r}}(x))$ is not an identically zero polynomial in the $\mathbf{r}$ variables and has degree less than $2n^2$, if we set every $\mathbf{r}$ variable uniformly and independently at random to a value in $[2n^3]$ then using Schwartz-Zippel lemma with probability at least $1 - \frac{1}{\mathrm{poly}(n)}$, $\gcd(h_{\mathbf{r}}, \frac{\partial h_{\mathbf{r}}}{\partial x}) = 1$. This implies with probability at least $1 - \frac{1}{\mathrm{poly}(n)}$, $h_{\mathbf{r}}(x)$ is square free. ◀

3.3 Invariant subspaces of $\mathfrak{g}_{\mathsf{IMM}}$

The ordering of the variables in IMM allows us to identify them naturally with the unit vectors $e_1, e_2, \ldots, e_n$ in $\mathbb{F}^n$ – the vector e_i corresponds to the i-th variable in the ordering. We will write e_x to refer to the unit vector corresponding to the variable x. Let $\mathcal{U}_{1,2}$ represent the coordinate subspace spanned by the unit vectors corresponding to the variables in $\mathbf{x}_1 \uplus \mathbf{x}_2$. Similarly $\mathcal{U}_k$ represents the coordinate subspace spanned by the unit vectors corresponding to the variables in $\mathbf{x}_k$ for $k \in [2, d-1]$, and $\mathcal{U}_{d-1,d}$ represents the coordinate subspace spanned by the unit vectors corresponding to the variables in $\mathbf{x}_{d-1} \uplus \mathbf{x}_d$. In Lemma 39, we establish that $\mathcal{U}_{1,2}, \mathcal{U}_2, \ldots, \mathcal{U}_{d-1}, \mathcal{U}_{d-1,d}$ are the only irreducible invariant subspaces of $\mathfrak{g}_{\mathsf{IMM}}$.

▶ **Claim 38.** *Let* $\mathcal{U}$ *be a nonzero invariant subspace of* $\mathfrak{g}_{\mathsf{IMM}}$. *If* $\mathbf{u} = (u_1, u_2, \ldots, u_n)^T \in \mathcal{U}$ *and* $u_j \neq 0$ *then* $e_j \in \mathcal{U}$, *implying* $\mathcal{U}$ *is a coordinate subspace.*

Proof. Claim 35 states that there is a diagonal matrix $D \in \mathfrak{g}_{\mathsf{IMM}}$ with distinct diagonal entries $\lambda_1, \lambda_2, \ldots, \lambda_n$. Since $\mathcal{U}$ is invariant for D, if $\mathbf{u} = (u_1, u_2, \ldots, u_n)^T \in \mathcal{U}$ then $(\lambda_1^i u_1, \lambda_2^i u_2, \ldots, \lambda_n^i u_n) \in \mathcal{U}$ for every $i \in \mathbb{N}$. Let $S_{\mathbf{u}} \coloneqq \{j \in [n] \mid u_j \neq 0\}$ be the support of $\mathbf{u} \neq 0$. As $\lambda_1, \lambda_2, \ldots, \lambda_n$ are distinct, the vectors $(\lambda_1^i u_1, \lambda_2^i u_2, \ldots, \lambda_n^i u_n)$ are $\mathbb{F}$-linearly independent for $0 \le i < |S_{\mathbf{u}}|$. Hence, the unit vector $e_j \in \mathcal{U}$ for every $j \in S_{\mathbf{u}}$. It follows that $\mathcal{U}$ is the coordinate subspace spanned by those e_j for which $j \in S_{\mathbf{u}}$ for some $\mathbf{u} \in \mathcal{U}$. ◀

▶ **Lemma 39.** *The only irreducible invariant subspaces of* $\mathfrak{g}_{\mathsf{IMM}}$ *are* $\mathcal{U}_{1,2}$, $\mathcal{U}_2$, …, $\mathcal{U}_{d-1}$, $\mathcal{U}_{d-1,d}$.

Proof. It follows from Lemma 31 and Figure 2 that $\mathcal{U}_{1,2}, \mathcal{U}_2, \ldots, \mathcal{U}_{d-1}, \mathcal{U}_{d-1,d}$ are invariant subspaces. We show in the next two claims that the spaces $\mathcal{U}_{1,2}, \mathcal{U}_2, \ldots, \mathcal{U}_{d-1}, \mathcal{U}_{d-1,d}$ are irreducible. The proofs are given in Section 7.2.

▶ **Claim 40.** *No invariant subspace of* $\mathfrak{g}_{\mathsf{IMM}}$ *is properly contained in* $\mathcal{U}_k$ *for* $k \in [2, d-1]$.

▶ **Claim 41.** *The invariant subspaces* $\mathcal{U}_{1,2}$ *and* $\mathcal{U}_{d-1,d}$ *are irreducible, and the only invariant subspace properly contained in* $\mathcal{U}_{1,2}$ *(respectively* $\mathcal{U}_{d-1,d}$*) is* $\mathcal{U}_2$ *(respectively* $\mathcal{U}_{d-1}$*).*

We in fact show in the proof of Claim 40 that the closure of e_x under the action of $\mathfrak{g}_{\mathsf{IMM}}$ is $\mathcal{U}_k$ for any $x \in \mathbf{x}_k$, where $k \in [2, d-1]$. Similarly, in the proof of Claim 41 we show that the closure of e_x under the action of $\mathfrak{g}_{\mathsf{IMM}}$ is $\mathcal{U}_{1,2}$ (respectively $\mathcal{U}_{d-1,d}$) for any $x \in \mathbf{x}_1$ (respectively $x \in \mathbf{x}_d$). This observation helps infer that the spaces $\mathcal{U}_{1,2}, \mathcal{U}_2, \ldots, \mathcal{U}_{d-1}, \mathcal{U}_{d-1,d}$ are the only irreducible invariant subspaces of $\mathfrak{g}_{\mathsf{IMM}}$: Suppose $\mathcal{V}$ is an irreducible invariant subspace. If $e_x \in \mathcal{V}$ for some $x \in \mathbf{x}_k$ where $k \in [2, d-1]$, then $\mathcal{U}_k \subseteq \mathcal{V}$ as $\mathcal{U}_k$ is the closure of e_x. If $e_x \in \mathcal{V}$ for some $x \in \mathbf{x}_1$ (respectively $x \in \mathbf{x}_d$) then $\mathcal{U}_{1,2} \subseteq \mathcal{V}$ (respectively $\mathcal{U}_{d-1,d} \subseteq \mathcal{V}$) as $\mathcal{U}_{1,2}$ (respectively $\mathcal{U}_{d-1,d}$) is the closure of e_x. Therefore $\mathcal{V}$ is a direct sum of some of the irreducible invariant subspaces $\mathcal{U}_{1,2}, \mathcal{U}_2, \ldots, \mathcal{U}_{d-1}, \mathcal{U}_{d-1,d}$. Since $\mathcal{V}$ is irreducible, it is equal to one of these irreducible invariant subspaces. ◀

▶ **Corollary 42** (Uniqueness of decomposition)**.** *The decomposition,*

$$\mathbb{F}^n = \mathcal{U}_{1,2} \oplus \mathcal{U}_3 \oplus \cdots \oplus \mathcal{U}_{d-2} \oplus \mathcal{U}_{d-1,d}$$

is unique in the following sense; if $\mathbb{F}^n = \mathcal{V}_1 \oplus \mathcal{V}_2 \oplus \cdots \oplus \mathcal{V}_s$*, where* $\mathcal{V}_i's$ *are irreducible invariant subspaces of* $\mathfrak{g}_{\mathsf{IMM}}$*, then* $s = d-2$ *and for every* $i \in [s]$*,* $\mathcal{V}_i$ *is equal to* $\mathcal{U}_{1,2}$ *or* $\mathcal{U}_{d-1,d}$*, or some* $\mathcal{U}_k$ *for* $k \in [3, d-2]$.

Proof. Since $\mathcal{V}_i$'s are irreducible invariant subspaces, from Lemma 39 it follows that for every $i \in [s]$ $\mathcal{V}_i$ equals one among $\mathcal{U}_{1,2}, \mathcal{U}_2, \ldots, \mathcal{U}_{d-1}, \mathcal{U}_{d-1,d}$. Since $\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_s$ span the entire $\mathbb{F}^n$, the only possible decomposition is $\mathbb{F}^n = \mathcal{U}_{1,2} \oplus \mathcal{U}_3 \oplus \cdots \oplus \mathcal{U}_{d-2} \oplus \mathcal{U}_{d-1,d}$. ◀

4 Lie algebra of f equivalent to IMM

Let f be an n variate polynomial such that $f = \mathsf{IMM}_{\mathbf{w},d}(A\mathbf{x})$, where $\mathbf{w} = (w_1, w_2, \ldots, w_{d-1}) \in \mathbb{N}^{d-1}$ and $A \in \mathsf{GL}(n)$. It follows, $n = w_1 + \sum_{i=2}^{d-1} w_{i-1}w_i + w_{d-1}$. From Observation 22 and Lemma 39 we know $A^{-1}\mathcal{U}_{1,2}$, $A^{-1}\mathcal{U}_2$, …, $A^{-1}\mathcal{U}_{d-1}$, $A^{-1}\mathcal{U}_{d-1,d}$ are the only irreducible invariant subspaces of $\mathfrak{g}_f$, and $A^{-1}\mathcal{U}_2$ (respectively $A^{-1}\mathcal{U}_{d-1}$) is the only invariant subspace properly contained in $A^{-1}\mathcal{U}_{1,2}$ (respectively $A^{-1}\mathcal{U}_{d-1,d}$). Also from Corollary 42 it follows that $\mathbb{F}^n = A^{-1}\mathcal{U}_{1,2} \oplus A^{-1}\mathcal{U}_3 \oplus \cdots \oplus A^{-1}\mathcal{U}_{d-2} \oplus A^{-1}\mathcal{U}_{d-1,d}$. In this section, we give an efficient randomized algorithm to compute a basis of each of the spaces $A^{-1}\mathcal{U}_{1,2}$, $A^{-1}\mathcal{U}_2$, …, $A^{-1}\mathcal{U}_{d-1}$, $A^{-1}\mathcal{U}_{d-1,d}$ given only blackbox access to f (but no knowledge of $\mathbf{w}$ or A).

4.1 Computing invariant subspaces of the Lie algebra $\mathfrak{g}_f$

First, we efficiently compute a basis $\{L_1', L_2', \ldots, L_m'\}$ of $\mathfrak{g}_f$ using the algorithm stated in Lemma 28. By Claim 21, $L_1 = AL_1'A^{-1}, L_2 = AL_2'A^{-1}, \ldots, L_m = AL_m'A^{-1}$ form a basis of $\mathfrak{g}_{\mathsf{IMM}}$. Suppose $R' = \sum_{i=1}^m r_i L_i'$ is a random element of $\mathfrak{g}_f$, chosen by picking the r_i's independently and uniformly at random from $[2n^3]$. Then $R = AR'A^{-1} = \sum_{i=1}^m r_i L_i$ is a

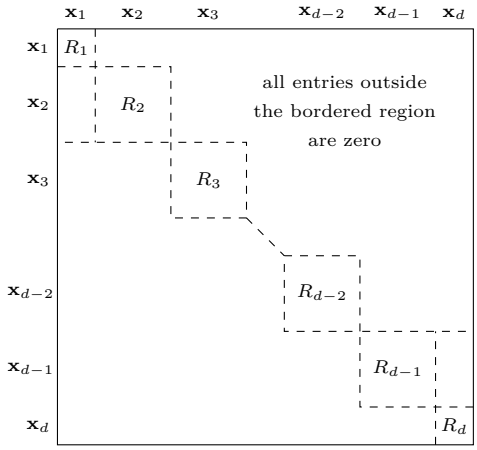


Figure 4 Random element R in $\mathfrak{g}_{\mathsf{IMM}}$.

random element of $\mathfrak{g}_{\mathsf{IMM}}$ and it follows from Lemma 36 that the characteristic polynomial of R is square free with probability at least $1 - \frac{1}{\text{poly}(n)}$. So assume henceforth that the characteristic polynomial of R (and hence also of $R^{'}$) is square free.

Moreover, from Figure 2 it follows that R has the structure as shown in Figure 4.

Let $h(x) = \prod_{i=1}^{d} h_i(x)$ be the characteristic polynomial of R and $R^{'}$, where $h_i(x)$ is the characteristic polynomial of R_i, and $g_1(x), g_2(x), \ldots, g_s(x)$ be the distinct irreducible factors of $h(x)$ over $\mathbb{F}$. Suppose $\mathcal{N}_i^{'}$ is the null space of $g_i(R^{'})$. Thus $\mathcal{N}_i$, the null space of $g_i(R)$ (equal to $A \cdot g_i(R^{'}) \cdot A^{-1}$), is $A\mathcal{N}_i^{'}$ for $i \in [s]$. We study the null spaces $\mathcal{N}_1, \mathcal{N}_2, \ldots, \mathcal{N}_s$ in the next two claims and show how to extract out the irreducible invariant subspaces of $\mathfrak{g}_f$ from $\mathcal{N}_1^{'}, \mathcal{N}_2^{'}, \ldots, \mathcal{N}_s^{'}$ (as specified in Algorithm 3). The proofs of these claims (using simple linear algebra) can be found in Section 7.3.

▶ **Claim 43.** *For all $i \in [s]$, let $\mathcal{N}_i$ and $\mathcal{N}_i^{'}$ be the null spaces of $g_i(R)$ and $g_i(R^{'})$. Then:*
1. $\mathbb{F}^n = \mathcal{N}_1 \oplus \mathcal{N}_2 \oplus \cdots \oplus \mathcal{N}_s = \mathcal{N}_1^{'} \oplus \mathcal{N}_2^{'} \oplus \cdots \oplus \mathcal{N}_s^{'}$.
2. *For all $i \in [s]$, $\dim(\mathcal{N}_i) = \dim(\mathcal{N}_i^{'}) = \deg_x(g_i)$.*

▶ **Claim 44.** *Suppose $g_i(x)$ is an irreducible factor of the characteristic polynomial $h_k(x)$ of R_k (depicted in Figure 4) for some $k \in [d]$. Then the following holds:*
1. *If $k \in [2, d-1]$ then $\mathcal{N}_i \subseteq \mathcal{U}_k$ (equivalently $\mathcal{N}_i^{'} \subseteq A^{-1}\mathcal{U}_k$).*
2. *If $k = 1$ then $\mathcal{N}_i \subseteq \mathcal{U}_{1,2}$ (equivalently $\mathcal{N}_i^{'} \subseteq A^{-1}\mathcal{U}_{1,2}$), and if $k = d$ then $\mathcal{N}_i \subseteq \mathcal{U}_{d-1,d}$ (equivalently $\mathcal{N}_i^{'} \subseteq A^{-1}\mathcal{U}_{d-1,d}$).*

▶ **Claim 45.**
1. *If $g_{l_1}(x), g_{l_2}(x), \ldots, g_{l_r}(x)$ are all the irreducible factors of $h_k(x)$ for $k \in [2, d-1]$ then $A^{-1}\mathcal{U}_k = \mathcal{N}_{l_1}^{'} \oplus \mathcal{N}_{l_2}^{'} \oplus \cdots \oplus \mathcal{N}_{l_r}^{'}$.*
2. *If $g_{l_1}(x), g_{l_2}(x), \ldots, g_{l_r}(x)$ are all the irreducible factors of $h_1(x)h_2(x)$ (respectively $h_{d-1}(x)$ $h_d(x)$) then $A^{-1}\mathcal{U}_{1,2} = \mathcal{N}_{l_1}^{'} \oplus \mathcal{N}_{l_2}^{'} \oplus \cdots \oplus \mathcal{N}_{l_r}^{'}$ (respectively $A^{-1}\mathcal{U}_{d-1,d} = \mathcal{N}_{l_1}^{'} \oplus \mathcal{N}_{l_2}^{'} \oplus \cdots \oplus \mathcal{N}_{l_r}^{'}$).*

Proof. If $k \in [2, d-1]$ then $\mathcal{N}_{l_1}^{'} + \mathcal{N}_{l_2}^{'} + \cdots + \mathcal{N}_{l_r}^{'}$ is a direct sum and

$$\dim(A^{-1}\mathcal{U}_k) = \deg_x(h_k) = \sum_{j=1}^{r} \deg_x(g_{l_j}) = \sum_{j=1}^{r} \dim(\mathcal{N}_{l_j}^{'}), \text{ which follow from Claim 43.}$$

Hence from Claim 44, $A^{-1}\mathcal{U}_k = \mathcal{N}_{l_1}^{'} \oplus \mathcal{N}_{l_2}^{'} \oplus \cdots \oplus \mathcal{N}_{l_r}^{'}$. The proof for the second part is similar. ◀

Algorithm 3 Computing irreducible invariant subspaces of $\mathfrak{g}_f$

INPUT: A basis $\{L'_1, L'_2, \ldots, L'_m\}$ of $\mathfrak{g}_f$.
OUTPUT: Bases of the irreducible invariant subspaces of $\mathfrak{g}_f$.

1. Pick a random element $R' = \sum_{j=1}^{m} r_j L'_j$ in $\mathfrak{g}_f$, where $r_j \in_R [2n^3]$.
2. Compute the characteristic polynomial $h(x)$ of R'.
3. **if** $h(x)$ is not square free **then**
4. Output 'Fail' and stop.
5. **end if**
6. Factor $h(x) = g_1(x) \cdot g_2(x) \ldots g_s(x)$ into irreducible factors over $\mathbb{F}$.
7. Find bases of the null spaces $\mathcal{N}'_1, \mathcal{N}'_2, \ldots, \mathcal{N}'_s$ of $g_1(R'), g_2(R'), \ldots, g_s(R')$ respectively.
8. For every $\mathcal{N}'_i$, pick a vector $\mathbf{v}$ in the basis of $\mathcal{N}'_i$ and compute the closure of $\mathbf{v}$ with respect to $\mathfrak{g}_f$ using Algorithm 4 given in Section 4.2.
9. Let $\{\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_s\}$ be the list of the closure spaces; check for all $i \neq j$ and $i, j \in [s]$, whether $\mathcal{V}_i = \mathcal{V}_j$ to remove repetitions from the above list and get the pruned list $\{\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_d\}$[31].
10. Output the set $\{\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_d\}$.

▶ **Lemma 46.** *Given as input bases of the null spaces $\mathcal{N}'_1, \mathcal{N}'_2, \ldots, \mathcal{N}'_s$ we can compute bases of the spaces $A^{-1}\mathcal{U}_{1,2}, A^{-1}\mathcal{U}_2, \ldots, A^{-1}\mathcal{U}_{d-1}, A^{-1}\mathcal{U}_{d-1,d}$ in deterministic polynomial time.*

Proof. Recall $\mathcal{N}'_i$ is the null space of $g_i(R')$, where $g_i(x)$ is an irreducible factor of $h_k(x)$ for some $k \in [d]$.

Case A: $k \in [2, d-1]$: From Claim 44 it follows that $\mathcal{N}'_i \subseteq A^{-1}\mathcal{U}_k$. Pick a basis vector $\mathbf{v}$ in $\mathcal{N}'_i$ and compute the closure of $\mathbf{v}$ under the action of $\mathfrak{g}_f$ using Algorithm 4 given in Section 4.2. Since the closure of $\mathbf{v}$ is the smallest invariant subspace of $\mathfrak{g}_f$ containing $\mathbf{v}$, by Claim 40 the closure of $\mathbf{v}$ equals $A^{-1}\mathcal{U}_k$.

Case B: $k = 1$ or $k = d$: The arguments for $k = 1$ and $k = d$ are similar. We prove it for $k = 1$. From Claim 44 we have $\mathcal{N}'_i \subseteq A^{-1}\mathcal{U}_{1,2}$. Pick a basis vector $\mathbf{v}$ of $\mathcal{N}'_i$ and compute its closure under the action of $\mathfrak{g}_f$ using Algorithm 4. Similar to case A, this gives us an invariant subspace of $\mathfrak{g}_f$ contained in $A^{-1}\mathcal{U}_{1,2}$ and by Claim 41 this invariant subspace is either $A^{-1}\mathcal{U}_2$ or $A^{-1}\mathcal{U}_{1,2}$. However, $\mathcal{N}'_i \cap A^{-1}\mathcal{U}_2$ (by Corollary 45) is empty, as $g_i(x)$ is an irreducible factor of $h_1(x)$ (not $h_2(x)$). Hence $\mathbf{v} \notin A^{-1}\mathcal{U}_2$ and the closure of $\mathbf{v}$ must be $A^{-1}\mathcal{U}_{1,2}$. ◀

To summarize, first we pick a random element R' in $\mathfrak{g}_f$, find its characteristic polynomial $h(x)$ and factorize $h(x)$ to get the irreducible factors $g_1(x), g_2(x), \ldots, g_s(x)$. Then we compute the null spaces $\mathcal{N}'_1, \mathcal{N}'_2, \ldots, \mathcal{N}'_s$ of $g_1(R'), g_2(R'), \ldots, g_s(R')$ respectively. By applying Claim 46, we find the invariant subspaces of $\mathfrak{g}_f$, $A^{-1}\mathcal{U}_{1,2}$, $A^{-1}\mathcal{U}_2$, …, $A^{-1}\mathcal{U}_{d-1}$, $A^{-1}\mathcal{U}_{d-1,d}$ from these null spaces. We present this formally in Algorithm 3.

Comments on Algorithm 3

(a) Observe that in step 6 of the algorithm we need $\mathbb{F}$ to be $\mathbb{Q}$ (as assumed) or a finite field because univariate factorization can be done effectively over such fields [31, 9, 11].

[31] Reusing symbols.

Algorithm 4 Computing the closure of $\mathbf{v}$ under the action of $\mathcal{L}$

INPUT: $\mathbf{v} \in \mathbb{F}^n$ and a basis $\{M_1, M_2, \ldots, M_m\}$ of $\mathcal{L}$.
OUTPUT: Basis of the closure of $\mathbf{v}$ under the action of $\mathcal{L}$.

1. Let $\mathcal{V}^{(0)} = \{\mathbf{v}\}$ and $\mathcal{V}^{(1)} = \text{span}_{\mathbb{F}}\{\mathbf{v}, M_1\mathbf{v}, \ldots, M_m\mathbf{v}\}$.
2. Set $i = 1$.
3. Compute a basis of $\mathcal{V}^{(1)}$ and let $T_1 = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{q_1}\}$ be this basis.
4. **while** $\mathcal{V}^{(i-1)} \neq \mathcal{V}^{(i)}$ **do**
5. Set $i = i + 1$.
6. Compute a basis for $\mathcal{V}^{(i)} = \text{span}_{\mathbb{F}}\{T_{i-1} \cup \mathcal{L} \cdot T_{i-1}\}$ and let $T_i = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{q_i}\}$ be this basis.
7. **end while**
8. Output T_i.

(b) When Algorithm 3 is invoked in Algorithm 2 for an n variate degree d polynomial f, there may not exists a $\mathbf{w} \in \mathbb{N}^{d-1}$ and an $A \in \mathsf{GL}(n)$ such that $f = \mathsf{IMM}_{\mathbf{w},d}(A\mathbf{x})$. We point out a few additional checks that need to be added to the above algorithm to handle this case. In step 9, if the pruned list (after removing repetitions) has size other than d then output 'Fail'. Also from Claim 41, exactly two subspaces in the pruned list $\{\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_d\}$, say $\mathcal{V}_2$ and $\mathcal{V}_{d-1}$, should be subspaces of other vector spaces, say $\mathcal{V}_1$ and $\mathcal{V}_d$ respectively. We can find these two spaces by doing a pairwise check among the d vector spaces. If such subspaces do not exist among $\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_d$ then output 'Fail'. Further, if $\mathbb{F}^n \neq \mathcal{V}_1 \oplus \mathcal{V}_3 \oplus \cdots \oplus \mathcal{V}_{d-2} \oplus \mathcal{V}_d$ (assuming $\mathcal{V}_2 \subseteq \mathcal{V}_1$ and $\mathcal{V}_{d-1} \subseteq \mathcal{V}_d$) then output 'Fail'.

(c) It follows from the above discussion , if $f = \mathsf{IMM}_{\mathbf{w},d}(A\mathbf{x})$ then we can assume $\mathcal{V}_3, \mathcal{V}_4, \ldots, \mathcal{V}_{d-2}$ are the spaces $A^{-1}\mathcal{U}_3, A^{-1}\mathcal{U}_4, \ldots, A^{-1}\mathcal{U}_{d-2}$ in some *unknown* order. The spaces $\mathcal{V}_1, \mathcal{V}_2$ and $\mathcal{V}_d, \mathcal{V}_{d-1}$ are either the spaces $A^{-1}\mathcal{U}_{1,2}, A^{-1}\mathcal{U}_2$ and $A^{-1}\mathcal{U}_{d-1,d}, A^{-1}\mathcal{U}_{d-1}$ respectively, or the spaces $A^{-1}\mathcal{U}_{d-1,d}$, $A^{-1}\mathcal{U}_{d-1}$ and $A^{-1}\mathcal{U}_{1,2}, A^{-1}\mathcal{U}_2$ respectively.

4.2 Closure of a vector under the action of $\mathfrak{g}_f$

Algorithm 4 computes the closure of $\mathbf{v} \in \mathbb{F}^n$ under the action of a space $\mathcal{L}$ spanned by $n \times n$ matrices. Let $\{M_1, M_2, \ldots, M_m\}$ be a basis of $\mathcal{L}$ where $M_i \in \mathbb{F}^{n \times n}$. For a set of vectors $T = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_q\} \subseteq \mathbb{F}^n$, let $\mathcal{L} \cdot T$ denote the set $\{M_a\mathbf{v}_b \mid a \in [m] \text{ and } b \in [q]\}$.

▶ **Claim 47.** *Algorithm 4 computes the closure of $\mathbf{v} \in \mathbb{F}^n$ under the action of $\mathcal{L}$ in time polynomial in n and the bit length of the entries of $\mathbf{v}$ and $M_1, M_2, \ldots, M_m$.*

Proof. The closure of $\mathbf{v}$ under the action of $\mathcal{L}$ is the $\mathbb{F}$-linear span of all vectors of the form $\mu.\mathbf{v}$, where μ is a non-commutative monomial in $M_1, M_2, \ldots, M_m$ (including unity). Algorithm 4 computes exactly this set and hence the closure of $\mathbf{v}$. Moreover, $\dim(\mathcal{V}^{(i)}) \leq n$ and in every iteration of the while loop $\dim(\mathcal{V}^{(i)}) > \dim(\mathcal{V}^{(i-1)})$, until $\mathcal{V}^{(i)} = \mathcal{V}^{(i-1)}$. Hence, Algorithm 4 runs in time polynomial in n and the bit length of the entries of $\mathbf{v}$ and $M_1, M_2, \ldots, M_m$. ◀

5 Reconstruction of full rank ABP for f

Let f be a polynomial equivalent to $\mathsf{IMM}_{\mathbf{w},d}$ for some (unknown) $\mathbf{w} \in \mathbb{N}^{d-1}$. In this section, we show that the invariant subspaces of $\mathfrak{g}_f$ let us compute a $\mathbf{w} \in \mathbb{N}^{d-1}$ and an $A \in \mathsf{GL}(n)$ such that $f = \mathsf{IMM}_{\mathbf{w},d}(A\mathbf{x})$. Since f is equivalent to $\mathsf{IMM}_{\mathbf{w},d}$, it is computable by a full rank ABP $X_1 \cdot X_2 \ldots X_{d-1} \cdot X_d$ of width $\mathbf{w}$ and length d with linear form entries in the

matrices. We call this full rank ABP $\mathtt{A}$ which, as explained below, is not the only full rank ABP computing f.

Many full rank ABPs for f: The full rank ABP $X'_1 \cdot X'_2 \cdots X'_d$ resulting from *each* of the following three transformations on $\mathtt{A}$ still computes f,

1. *Transposition*: Set $X'_k = X^T_{d+1-k}$ for $k \in [d]$.
2. *Left-right multiplications*: Let $A_1, \ldots, A_{d-1}$ be matrices such that $A_k \in \mathsf{GL}(w_k)$ for every $k \in [d-1]$. Set $X'_1 = X_1 \cdot A_1$, $X'_d = A^{-1}_{d-1} \cdot X_d$, and $X'_k = A^{-1}_{k-1} \cdot X_k \cdot A_k$ for $k \in [2, d-1]$.
3. *Corner translations*: Suppose $\{C_{11}, C_{12}, \ldots, C_{1w_2}\}$ and $\{C_{d1}, C_{d2}, \ldots, C_{dw_{d-2}}\}$ are two sets containing anti-symmetric matrices in $\mathbb{F}^{w_1 \times w_1}$ and $\mathbb{F}^{w_{d-1} \times w_{d-1}}$ respectively. Let $Y_2 \in \mathbb{F}[\mathbf{x}]^{w_1 \times w_2}$ (respectively $Y_{d-1} \in \mathbb{F}[\mathbf{x}]^{w_{d-2} \times w_{d-1}}$) be a matrix with its i-th column (respectively i-th row) equal to $C_{1i} \cdot X^T_1$ (respectively $X^T_d \cdot C_{di}$). Set $X'_2 = X_2 + Y_2$, $X'_{d-1} = X_{d-1} + Y_{d-1}$, and $X'_k = X_k$ for $k \in [d] \setminus \{2, d-1\}$.

In each of the above three cases $f = X'_1 \cdot X'_2 \cdots X'_d$; this is easy to verify for cases 1 and 2, in case 3 observe that $X_1 \cdot C_{1i} \cdot X^T_1 = X^T_d \cdot C_{di} \cdot X_d = 0$.

It turns out that the full rank ABPs obtained by (repeatedly) applying the above three transformations on $\mathtt{A}$ are the only full rank ABPs computing f. This would follow from the discussion in Section 6. Although there are multiple full rank ABPs for f, the layer spaces of these ABPs are unique (Lemma 48). This uniqueness of the layer spaces essentially facilitates the recovery of a full rank ABP for f. Let us denote the span of the linear forms[32] in X_1 and X_2 (respectively X_{d-1} and X_d) by $\mathcal{X}_{1,2}$ (respectively $\mathcal{X}_{d-1,d}$).

▶ **Lemma 48** (Uniqueness of the layer spaces of full rank ABP for f)**.** *Suppose $X_1 \cdot X_2 \cdots X_d$ and $X'_1 \cdot X'_2 \cdots X'_d$ are two full rank ABPs of widths $\mathbf{w} = (w_1, w_2, \ldots, w_{d-1})$ and $\mathbf{w}' = (w'_1, w'_2, \ldots, w'_{d-1})$ respectively, computing the same polynomial f. Then one of the following two cases is true:*

(a) *$w'_k = w_k$ for $k \in [d-1]$, and the spaces $\mathcal{X}'_1, \mathcal{X}'_{1,2}, \mathcal{X}'_3, \ldots, \mathcal{X}'_{d-1,d}, \mathcal{X}'_d$ are the spaces $\mathcal{X}_1, \mathcal{X}_{1,2}, \mathcal{X}_3, \ldots, \mathcal{X}_{d-1,d}, \mathcal{X}_d$ respectively.*

(b) *$w'_k = w_{d-k}$ for $k \in [d-1]$, and the spaces $\mathcal{X}'_1, \mathcal{X}'_{1,2}, \mathcal{X}'_3, \ldots, \mathcal{X}'_{d-1,d}, \mathcal{X}'_d$ are the spaces $\mathcal{X}_d, \mathcal{X}_{d-1,d}, \mathcal{X}_{d-2}, \ldots, \mathcal{X}_{1,2}, \mathcal{X}_1$ respectively.*

The lemma would help characterize the group of symmetries of IMM in Section 6; the proof would follow readily from Claim 50 in Section 5.2. With an eye on Section 6 and for better clarity in the reduction to almost set-multilinear ABP in Section 5.2, we take a slight detour and show next how to compute these 'unique' layer spaces of $\mathtt{A}$.

5.1 Computing layer spaces from invariant subspaces of $\mathfrak{g}_f$

Algorithm 3 outputs bases of the irreducible invariant subspaces $\{\mathcal{V}_i \mid i \in [d]\}$ of $\mathfrak{g}_f$. Recall, we assumed without loss of generality that $\mathcal{V}_2$ and $\mathcal{V}_{d-1}$ are subspaces of $\mathcal{V}_1$ and $\mathcal{V}_d$ respectively. The spaces $\mathcal{V}_1, \mathcal{V}_2$ and $\mathcal{V}_d, \mathcal{V}_{d-1}$ are either the spaces $A^{-1}\mathcal{U}_{1,2}, A^{-1}\mathcal{U}_2$ and $A^{-1}\mathcal{U}_{d-1,d}, A^{-1}\mathcal{U}_{d-1}$ respectively, or the spaces $A^{-1}\mathcal{U}_{d-1,d}, A^{-1}\mathcal{U}_{d-1}$ and $A^{-1}\mathcal{U}_{1,2}, A^{-1}\mathcal{U}_2$ respectively. Every other $\mathcal{V}_k$ is equal to $A^{-1}\mathcal{U}_{\sigma(k)}$ for some permutation σ on $[3, d-2]$ (σ is not known at the end of Algorithm 3). Hence,

$$\mathbb{F}^n = \mathcal{V}_1 \oplus \mathcal{V}_3 \oplus \cdots \oplus \mathcal{V}_{d-2} \oplus \mathcal{V}_d. \tag{3}$$

[32] Identify linear forms with vectors in $\mathbb{F}^n$ as mentioned in Definition 15.

Algorithm 5 Computing the layer spaces of $\mathtt{A}$

INPUT: Bases of the irreducible invariant subspaces of $\mathfrak{g}_f$.
OUTPUT: Bases of the layer spaces of $\mathtt{A}$.

1. Form an $n \times n$ matrix V by concatenating the columns of the matrices $V_1, V_3, \ldots, V_{d-2}, V_d$ in order, that is $V = [V_1 \mid V_3 \mid \ldots \mid V_{d-2} \mid V_d]$.
2. Compute V^{-1}. Number the rows of V^{-1} by 1 to n.
3. Let $\mathcal{Y}_1$ be the space spanned by the first $u_1 - u_2$ rows of V^{-1}, and $\mathcal{Y}_{1,2}$ be the space spanned by the first u_1 rows of V^{-1}. Let $\mathcal{Y}_{d-1,d}$ be the space spanned by the last u_d rows of V^{-1} and $\mathcal{Y}_d$ be the space spanned by the last $u_d - u_{d-1}$ rows of V^{-1}. Finally, for every $k \in [3, d-2]$, let $\mathcal{Y}_k$ be the space spanned by the rows of V^{-1} that are numbered by $t_{k-1}+1$ to $t_{k-1}+u_k$. Output the bases of the spaces $\mathcal{Y}_1, \mathcal{Y}_{1,2}, \mathcal{Y}_3, \ldots, \mathcal{Y}_{d-2}, \mathcal{Y}_{d-1,d}, \mathcal{Y}_d$ in order.

Since $\mathcal{V}_2 \subseteq \mathcal{V}_1$, we can start with a basis of $\mathcal{V}_2$ and fill in more elements from the basis of $\mathcal{V}_1$ to get a new basis of $\mathcal{V}_1$. Thus we can assume the basis of $\mathcal{V}_2$ is contained in the basis of $\mathcal{V}_1$. Likewise, the basis of $\mathcal{V}_{d-1}$ is contained in the basis of $\mathcal{V}_d$.

Order the basis vectors of $\mathcal{V}_1$ such that the basis vectors of $\mathcal{V}_2$ are at the end and order the basis vectors of $\mathcal{V}_d$ such that the basis vectors of $\mathcal{V}_{d-1}$ are at the beginning. For $k \in [3, d-2]$, the basis vectors of $\mathcal{V}_k$ are ordered in an arbitrary way. Let u_k denote the dimension of $\mathcal{V}_k$ for $k \in [d]$. We identify the space $\mathcal{V}_k$ with an $n \times u_k$ matrix V_k, where the i-th column in V_k is the i-th basis vector of $\mathcal{V}_k$ in the above specified order. Algorithm 5 computes the layer spaces of $\mathtt{A}$ using V_1 to V_d. Let $t_2 = u_1$ and $t_k = u_k + t_{k-1}$ for $k \in [3, d-2]$.

Comments on Algorithm 5: Algorithm 2 invokes Algorithm 5 only after Algorithm 3, which returns 'Fail' if $\mathbb{F}^n \neq \mathcal{V}_1 \oplus \mathcal{V}_3 \oplus \cdots \oplus \mathcal{V}_{d-2} \oplus \mathcal{V}_d$ (see comments after Algorithm 3). This ensures Equation (3) is satisfied and so V^{-1} exists in step 2 of the above algorithm, even if there are no $\mathbf{w} \in \mathbb{N}^{d-1}$ and $A \in \mathsf{GL}(n)$ such that $f = \mathsf{IMM}_{\mathbf{w},d}(A\mathbf{x})$.

▶ **Lemma 49.** *If $f = X_1 \cdot X_2 \cdots X_d$ and $\mathcal{Y}_1, \mathcal{Y}_{1,2}, \mathcal{Y}_3, \ldots, \mathcal{Y}_{d-2}, \mathcal{Y}_{d-1,d}, \mathcal{Y}_d$ is the output of Algorithm 5 then there is a permutation σ on $[3, d-2]$ such that the following hold:*

1. *For every $k \in [3, d-2]$, $\mathcal{Y}_k = \mathcal{X}_{\sigma(k)}$.*
2. *Either $\mathcal{Y}_1, \mathcal{Y}_{1,2}$ and $\mathcal{Y}_d, \mathcal{Y}_{d-1,d}$ are $\mathcal{X}_1, \mathcal{X}_{1,2}$ and $\mathcal{X}_d, \mathcal{X}_{d-1,d}$ respectively, or $\mathcal{Y}_1, \mathcal{Y}_{1,2}$ and $\mathcal{Y}_d$, $\mathcal{Y}_{d-1,d}$ are $\mathcal{X}_d, \mathcal{X}_{d-1,d}$ and $\mathcal{X}_1, \mathcal{X}_{1,2}$ respectively.*

The proof is given in Section 7.4.

5.2 Reduction to almost set-multilinear ABP

The outline: Once the invariant spaces of $\mathfrak{g}_f$ are computed, the reduction proceeds like this: As observed in the proof of Lemma 49, the matrix V in Algorithm 5 equals $A^{-1}E$ where E looks as shown in Figure 14. If $f = \mathsf{IMM}_{\mathbf{w},d}(A\mathbf{x})$ then $f(V\mathbf{x}) = \mathsf{IMM}_{\mathbf{w},d}(E\mathbf{x})$. Owing to the structure of E, $f(V\mathbf{x})$ is computed by a full rank almost set-multilinear ABP, except that the ordering of the groups of variables occurring in the different layers of the ABP is unknown as σ is unknown. The 'correct' ordering along with a width vector can be retrieved by applying evaluation dimension, thereby completing the reduction. For a slightly neater presentation of the details (and with the intent of proving Lemma 48), we deviate from this strategy a little bit and make use of the layer spaces that have already been computed by Algorithm 5.

Algorithm 6 Reduction to full rank almost set-multilinear ABP

INPUT: Bases of the layer spaces $\mathcal{Y}_1, \mathcal{Y}_{1,2}, \mathcal{Y}_3, \ldots, \mathcal{Y}_{d-2}, \mathcal{Y}_{d-1,d}, \mathcal{Y}_d$ from Algorithm 5.
OUTPUT: A $\mathbf{w} \in \mathbb{N}^{d-1}$ and an $\widehat{A} \in \mathsf{GL}(n)$ such that $f(\widehat{A}\mathbf{x})$ is computable by a full rank almost set-multilinear ABP of width $\mathbf{w}$.

1. Reorder the layer spaces to $\mathcal{X}_1, \mathcal{X}_{1,2}, \mathcal{X}_3, \ldots, \mathcal{X}_{d-2}, \mathcal{X}_{d-1,d}, \mathcal{X}_d$ and obtain $\mathbf{w}$ (using Claim 50). /* This step succeeds with high probability if f is equivalent to $\mathsf{IMM}_{\mathbf{w},d}$ for some $\mathbf{w}$. */
2. Find $\widehat{A} \in \mathsf{GL}(n)$ from the reordered spaces and $\mathbf{w}$ (using Claim51).

The details: Algorithm 5 computes the spaces $\mathcal{Y}_1, \mathcal{Y}_{1,2}, \mathcal{Y}_3, \ldots, \mathcal{Y}_{d-2}, \mathcal{Y}_{d-1,d}, \mathcal{Y}_d$ which (according to Lemma 49) are either the spaces $\mathcal{X}_1, \mathcal{X}_{1,2}, \mathcal{X}_{\sigma(3)}, \ldots, \mathcal{X}_{\sigma(d-2)}, \mathcal{X}_{d-1,d}, \mathcal{X}_d$ respectively, or the spaces $\mathcal{X}_d, \mathcal{X}_{d-1,d}, \mathcal{X}_{\sigma(3)}, \ldots, \mathcal{X}_{\sigma(d-2)}, \mathcal{X}_{1,2}, \mathcal{X}_1$ respectively, for some unknown permutation σ on $[3, d-2]$. The claim below (proved in Section 7.4) shows how to correctly reorder these layer spaces.

▶ **Claim 50.** *There is a randomized polynomial time algorithm that takes input the bases of the layer spaces $\mathcal{Y}_1, \mathcal{Y}_{1,2}, \mathcal{Y}_3, \ldots, \mathcal{Y}_{d-2}, \mathcal{Y}_{d-1,d}, \mathcal{Y}_d$ and with probability at least $1 - \frac{1}{\mathrm{poly}(n)}$ reorders these layer spaces and outputs a width vector $\mathbf{w}'$ such that the reordered sequence of spaces and $\mathbf{w}'$ are:*
1. *either $\mathcal{X}_1, \mathcal{X}_{1,2}, \mathcal{X}_3, \ldots, \mathcal{X}_{d-2}, \mathcal{X}_{d-1,d}, \mathcal{X}_d$ and $(w_1, w_2, \ldots, w_{d-1})$ respectively,*
2. *or $\mathcal{X}_d, \mathcal{X}_{d-1,d}, \mathcal{X}_{d-2}, \ldots, \mathcal{X}_3, \mathcal{X}_{1,2}, \mathcal{X}_1$ and $(w_d, w_{d-1}, \ldots, w_1)$ respectively.*

Note: Until the algorithm in the claim is applied to reorder the spaces, Algorithm 2 is totally oblivious of the width vector $\mathbf{w}$ (it has been used only in the analysis thus far). So, due to the legitimacy of the transposition transformation mentioned at the beginning of this section, we may as well assume that the $\mathbf{w}'$ in the above claim is in fact our $\mathbf{w}$, and the output ordered sequence of spaces is $\mathcal{X}_1, \mathcal{X}_{1,2}, \mathcal{X}_3, \ldots, \mathcal{X}_{d-2}, \mathcal{X}_{d-1,d}, \mathcal{X}_d$.

▶ **Claim 51.** *Given bases of the spaces $\mathcal{X}_1, \mathcal{X}_{1,2}, \mathcal{X}_3, \ldots, \mathcal{X}_{d-2}, \mathcal{X}_{d-1,d}, \mathcal{X}_d$ and $\mathbf{w}$, we can find an $\widehat{A} \in \mathsf{GL}(n)$ in polynomial time such that $f(\widehat{A}\mathbf{x})$ is computable by a full rank almost set-multilinear ABP of width $\mathbf{w}$.*

Proof. Identify the variables $x_1, \ldots, x_n$ with the variables $\mathbf{x}_1 \uplus \ldots \uplus \mathbf{x}_d$ of $\mathsf{IMM}_{\mathbf{w},d}$ following the ordering prescribed in Section 2.3. The map $\mathbf{x} \mapsto \widehat{A}\mathbf{x}$ should satisfy the following conditions:

(a) For every $k \in [3, d-2]$, the linear forms corresponding[33] to the basis vectors of $\mathcal{X}_k$ map to distinct variables in $\mathbf{x}_k$.
(b) The linear forms corresponding to the basis vectors in $\mathcal{X}_1$ (similarly, $\mathcal{X}_d$) map to distinct variables in $\mathbf{x}_1$ (similarly, $\mathbf{x}_d$).
(c) The linear forms corresponding to the basis vectors in $\mathcal{X}_{1,2}$ (similarly, $\mathcal{X}_{d-1,d}$) map to distinct variables in $\mathbf{x}_1 \uplus \mathbf{x}_2$ (similarly, $\mathbf{x}_{d-1} \uplus \mathbf{x}_d$).

Conditions (b) and (c) can be simultaneously satisfied as the basis of $\mathcal{X}_1$ (similarly, $\mathcal{X}_d$) is contained in the basis of $\mathcal{X}_{1,2}$ (similarly, $\mathcal{X}_{d-1,d}$) by construction. Such an $\widehat{A}$ can be easily obtained. ◀

We summarize the discussion in Algorithm 6.

[33] Recall, linear forms in $\mathbf{x}$ variables and vectors in $\mathbb{F}^n$ are naturally identified with each other.

Comments on Algorithm 6: The proof of Claim 50 includes Observation 67 which helps Algorithm 6 in step 1 to reorder the layer spaces. If f is not equivalent to $\mathsf{IMM}_{\mathbf{w},d}$ for some $\mathbf{w}$ then Algorithm 6 may fail in step 1, as at some stage it may not be able to find a variable set $\mathbf{z}_k$ such that $\text{Evaldim}_{\mathbf{y}_j \uplus \mathbf{z}_k}(h) < |\mathbf{z}_k|$ (see proof of Observation 67). When Algorithm 2 invokes Algorithm 6, if step 1 fails then the latter outputs 'Fail' and stops.

5.3 Reconstructing almost set-multilinear ABP

We prove Claim 30 in this section. Let $h = f(\widehat{A}\mathbf{x})$; identify $\mathbf{x}$ with the variables $\mathbf{x}_1 \uplus \ldots \uplus \mathbf{x}_d$ of $\mathsf{IMM}_{\mathbf{w},d}$ as before. From Claim 51, h is computable by a full rank almost set-multilinear ABP of width $\mathbf{w}$. Algorithm 2 uses Algorithm 7 to reconstruct a full rank almost set-multilinear ABP for h and then it replaces $\mathbf{x}$ by $\widehat{A}^{-1}\mathbf{x}$ to output a full rank ABP for f. The correctness of Algorithm 7 is presented as part of the proof of Claim 30. We begin with the following two observations the proofs of which appear in Section 7.4.

▶ **Observation 52.** *If h is computable by a full rank almost set-multilinear ABP of width $\mathbf{w}$ then there is a full rank almost set-multilinear ABP of width $\mathbf{w}$ in canonical form computing h.*

▶ **Observation 53.** *Let $X_1 \cdot X_2 \cdots X_d$ be a full rank almost set-multilinear ABP, and $C_k = X_k \cdots X_d$ for $k \in [2, d]$. Let the l-th entry of C_k be h_{kl} for $l \in [w_{k-1}]$. Then the polynomials $\{h_{k1}, h_{k2}, \cdots, h_{kw_{k-1}}\}$ are $\mathbb{F}$-linearly independent.*

Notations for Algorithm 7: For $k \in [d-1]$, let $t_k = |\mathbf{x}_1 \uplus \mathbf{x}_2 \uplus \cdots \uplus \mathbf{x}_k|$ and $m_k = |\mathbf{x}_{k+1} \uplus \mathbf{x}_{k+2} \uplus \cdots \uplus \mathbf{x}_d|$. The (i,j)-th entry of a matrix X is denoted by $X(i,j)$, and $e_{w_k,i}$ denotes a vector in $\mathbb{F}^{w_k}$ with the i-th entry 1 and other entries 0. Let $\mathbf{y}_i$ denote the following partial assignment to the $\mathbf{x}_1$ variables: $x_i^{(1)}, \ldots, x_{w_1}^{(1)}$ are kept intact, while the remaining variables are set to zero. Similarly, $\mathbf{z}_j$ denotes the following partial assignment to the $\mathbf{x}_d$ variables: $x_j^{(d)}, \ldots, x_{w_{d-1}}^{(d)}$ are kept intact, while the remaining variables are set to zero. The notation $h(\mathbf{a}_i, \mathbf{x}_k, \mathbf{b}_j)$ means the variables $\mathbf{x}_1 \uplus \ldots \uplus \mathbf{x}_{k-1}$ are given the assignment $\mathbf{a}_i \in \mathbb{F}^{t_{k-1}}$ and the variables $\mathbf{x}_{k+1} \uplus \ldots \uplus \mathbf{x}_d$ are given the assignment $\mathbf{b}_j \in \mathbb{F}^{m_k}$. The connotations for $h(\mathbf{y}_i, \mathbf{x}_2, \mathbf{b}_j)$ and $h(\mathbf{a}_i, \mathbf{x}_{d-1}, \mathbf{z}_j)$ are similar. The function $\text{poly}(n)$ is a suitably large polynomial function in n, say n^7.

Proof of Claim 30. By Observation 52, there is a full rank ABP $X_1' \cdot X_2' \cdots X_d'$ in canonical form computing h. Hence $X_1 = X_1' = (x_1^{(1)}\ x_2^{(1)}\ \ldots\ x_{w_1}^{(1)})$ and $X_d = X_d' = (x_1^{(d)}\ x_2^{(d)}\ \ldots\ x_{w_{d-1}}^{(d)})$. We show next that with probability at least $1 - \frac{1}{\text{poly}(n)}$, Algorithm 7 constructs $X_2, X_3, \ldots, X_{d-1}$ such that $X_2 = X_2' \cdot T_2$, $X_{d-1} = T_{d-2}^{-1} \cdot X_{d-1}'$ and $X_k = T_{k-1}^{-1} \cdot X_k' \cdot T_k$ for every $k \in [3, d-2]$, where $T_i \in \mathsf{GL}(w_i)$ for $i \in [2, d-2]$.

Steps 3–13: The matrix X_2 is formed in these steps. By Observation 53, the polynomials $h_{31}, \ldots, h_{3w_2}$ are $\mathbb{F}$-linearly independent. Since $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_{w_2}$ are randomly chosen in step 3, the matrix T_2 with (r,c)-th entry $h_{3r}(\mathbf{b}_c)$ is in $\mathsf{GL}(w_2)$ with high probability. Let $X_2'T_2(i,j)$ be the (i,j)-th entry of $X_2'T_2$. Observe that

$$h(\mathbf{y}_i, \mathbf{x}_2, \mathbf{b}_j) = X_2'T_2(i,j) \cdot x_i^{(1)} + \ldots + X_2'T_2(w_1,j) \cdot x_{w_1}^{(1)}.$$

As $h(\mathbf{y}_i, \mathbf{x}_2, \mathbf{b}_j)$ is a quadratic polynomial, we can compute it from blackbox access using the sparse polynomial interpolation algorithm in [30]. By induction on the rows, $X_2(p,j) = X_2'T_2(p,j)$ for every $p \in [i+1, w_1]$ and $j \in [w_2]$. So in step 8, $g_j = X_2'T_2(i,j) \cdot x_i^{(1)}$ leading to $X_2(i,j) = X_2'T_2(i,j)$ in step 9.

Algorithm 7 Reconstruction of full rank almost set-multlinear ABP

INPUT: Blackbox access to an n variate polynomial h and the width vector $\mathbf{w}$.
OUTPUT: A full rank almost set-multilinear ABP of width $\mathbf{w}$ in canonical form computing h.

Set $X_1 = (x_1^{(1)}\ x_2^{(1)}\ \dots\ x_{w_1}^{(1)})$ and $X_d = (x_1^{(d)}\ x_2^{(d)}\ \dots\ x_{w_{d-1}}^{(d)})^T$.

Choose w_2 random points $\{\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_{w_2}\}$ from S^{m_2} such that $S \subset \mathbb{F}$ and $|S| = \text{poly}(n)$.

Set $i = w_1$.
while $i \geq 1$ **do**
  for every $j \in [w_2]$ **do**
    Interpolate the quadratic $h(\mathbf{y}_i, \mathbf{x}_2, \mathbf{b}_j)$.
    Set $g_j = h(\mathbf{y}_i, \mathbf{x}_2, \mathbf{b}_j) - \sum_{p=i+1}^{w_1} X_2(p,j) \cdot x_p^{(1)}$.
    If g_j is not divisible by $x_i^{(1)}$, output 'Fail'. Else, set $X_2(i,j) = g_j / x_i^{(1)}$.
  end for
  Set $i = i - 1$.
end while
If the linear forms in X_2 are not $\mathbb{F}$-linearly independent, output 'Fail'.

Set $k = 3$.
while $k \leq d-2$ **do**
  Find w_{k-1} evaluations, $\{\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_{w_{k-1}}\} \subset \mathbb{F}^{t_{k-1}}$, of $\mathbf{x}_1 \uplus \mathbf{x}_2 \uplus \dots \uplus \mathbf{x}_{k-1}$ variables such that $X_1 \cdot X_2 \cdots X_{k-1}$ evaluated at $\mathbf{a}_i$ equals $e_{w_{k-1},i}$.
  Choose w_k random points $\{\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_{w_k}\}$ from S^{m_k} such that $S \subset \mathbb{F}$ and $|S| = \text{poly}(n)$.
  Interpolate the linear forms $h(\mathbf{a}_i, \mathbf{x}_k, \mathbf{b}_j)$ for $i \in [w_{k-1}], j \in [w_k]$.
  Set $X_k(i,j) = h(\mathbf{a}_i, \mathbf{x}_k, \mathbf{b}_j)$ for $i \in [w_{k-1}]$, $j \in [w_k]$.
  If the linear forms in X_k are not $\mathbb{F}$-linearly independent, output 'Fail'.
  Set $k = k + 1$.
end while

Find w_{d-2} evaluations, $\{\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_{w_{d-2}}\} \subset \mathbb{F}^{t_{d-2}}$, of $\mathbf{x}_1 \uplus \mathbf{x}_2 \uplus \dots \uplus \mathbf{x}_{d-2}$ variables such that $X_1 \cdot X_2 \cdots X_{d-2}$ evaluated at $\mathbf{a}_i$ equals $e_{w_{d-2},i}$.
Set $j = w_{d-1}$.
while $j \geq 1$ **do**
  for every $i \in [w_{d-2}]$ **do**
    Interpolate the quadratic $h(\mathbf{a}_i, \mathbf{x}_{d-1}, \mathbf{z}_j)$.
    Set $g_i = h(\mathbf{a}_i, \mathbf{x}_{d-1}, \mathbf{z}_j) - \sum_{q=j+1}^{w_{d-1}} X_{d-1}(i,q) \cdot x_q^{(d)}$.
    If g_i is not divisible by $x_j^{(d)}$, output 'Fail'. Else, set $X_{d-1}(i,j) = g_i / x_j^{(d)}$.
  end for
  Set $j = j - 1$.
end while
If the linear forms in X_{d-1} are not $\mathbb{F}$-linearly independent, output 'Fail'.

Output $X_1 \cdot X_2 \cdots X_{d-1} \cdot X_d$ as the full rank almost set-multilinear ABP for h.

Steps 15–23: The matrices $X_3, \ldots, X_{d-2}$ are formed in these steps. By the time the algorithm reaches step 17, it has already computed $X_2, \ldots, X_{k-1}$ such that $X_2 = X_2'T_2$ and $X_q = T_{q-1}^{-1}X_q'T_q$ for $q \in [3, k-1]$, where $T_q \in \mathsf{GL}(w_q)$. So, $X_1' \ldots X_{k-1}' = X_1 \ldots X_{k-1}T_{k-1}^{-1}$. As the linear forms in $X_1, \ldots, X_{k-1}$ are $\mathbb{F}$-linearly independent (otherwise the algorithm would have terminated in step 13 or 21), we can easily compute points $\{\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_{w_{k-1}}\}$ satisfying the required condition in step 17. By Observation 53, the polynomials $h_{(k+1)1}, \ldots, h_{(k+1)w_k}$ are $\mathbb{F}$-linearly independent. Since $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_{w_k}$ are randomly chosen in step 18, the matrix T_k with (r, c)-th entry $h_{(k+1)r}(\mathbf{b}_c)$ is in $\mathsf{GL}(w_k)$ with high probability. Now observe that $h(\mathbf{a}_i, \mathbf{x}_k, \mathbf{b}_j)$ is the (i, j)-th entry of $T_{k-1}^{-1}X_k'T_k$, which implies $X_k = T_{k-1}^{-1}X_k'T_k$ from step 20.

Steps 25–35: In these steps, matrix X_{d-1} is formed. The argument showing $X_{d-1} = T_{d-2}^{-1}X_{d-1}'$ is similar to the argument used for steps 3–13, except that now we induct on columns instead of rows.

The output ABP $X_1 \ldots X_d$ is in canonical form as $X_1' \ldots X_d'$ is also in canonical form. It is clear that the total running time of the algorithm is $\mathrm{poly}(n, \beta)$, where β is the bit length of the coefficients of h which influences the bit length of the values returned by the blackbox. ◀

6 Symmetries of IMM

Recall from Section 2.3, $\mathsf{IMM}_{\mathbf{w},d}$ (for brevity IMM) is the n variate polynomial computed by the full rank ABP $Q_1 \cdot Q_2 \cdots Q_d$ where the set of variables in Q_k is $\mathbf{x}_k$ for every $k \in [d]$. In this section, we determine the group of symmetries of IMM (denoted by $\mathcal{G}_{\mathsf{IMM}}$) and show that IMM is characterized by its symmetries. We make a note of a few notations and terminologies below.

Notations

- Calligraphic letters $\mathcal{H}$, $\mathcal{C}$, $\mathcal{M}$ and $\mathcal{T}$ denote subgroups of $\mathcal{G}_{\mathsf{IMM}}$. Let $\mathcal{C}$ and $\mathcal{H}$ be subgroups of $\mathcal{G}_{\mathsf{IMM}}$ such that $\mathcal{C} \cap \mathcal{H} = I_n$ and for every $H \in \mathcal{H}$ and $C \in \mathcal{C}$, $H \cdot C \cdot H^{-1} \in \mathcal{C}$. Then $\mathcal{C} \rtimes \mathcal{H}$ denotes the *semidirect product* of $\mathcal{C}$ and $\mathcal{H}$[34].
- For every $A \in \mathcal{G}_{\mathsf{IMM}}$ the full rank ABP obtained by replacing $\mathbf{x}$ by $A\mathbf{x}$ in $Q_1 \cdot Q_2 \cdots Q_d$ is termed as the full rank ABP *determined by* A. This full rank ABP also computes IMM.
- Let X be a matrix with entries as linear forms in $\mathbf{y} \uplus \mathbf{z}$ variables. We break X into two parts $X(\mathbf{y})$ and $X(\mathbf{z})$ such that $X = X(\mathbf{y}) + X(\mathbf{z})$. The (i, j)-th linear form in $X(\mathbf{y})$ (respectively $X(\mathbf{z})$) is the part of the (i, j)-th linear form of X in $\mathbf{y}$ (respectively $\mathbf{z}$) variables.

6.1 The group $\mathcal{G}_{\mathsf{IMM}}$

Three subgroups of $\mathcal{G}_{\mathsf{IMM}}$: As before, let $\mathbf{w} = (w_1, w_2, \ldots, w_{d-1})$ and $w_k > 1$ for every $k \in [d-1]$. In Theorem 54 below, we show that $\mathcal{G}_{\mathsf{IMM}}$ is generated by three special subgroups.

1. *Transposition subgroup* $\mathcal{T}$: If $w_k \neq w_{d-k}$ for any $k \in [d-1]$ then $\mathcal{T}$ is the trivial group containing only I_n. Otherwise, if $w_k = w_{d-k}$ for every $k \in [d-1]$ then $\mathcal{T}$ is the group consisting of two elements I_n and T. The matrix T is such that the full rank ABP determined by T is $Q_d^T \cdot Q_{d-1}^T \cdots Q_1^T$. Clearly, T is a permutation matrix and $T^2 = I_n$.

[34] $\mathcal{C} \rtimes \mathcal{H}$ is the set $\mathcal{C}\mathcal{H}$ which can be easily shown to be a subgroup of $\mathcal{G}_{\mathsf{IMM}}$, and it also follows that $\mathcal{C}$ is a normal subgroup of $\mathcal{C} \rtimes \mathcal{H}$.

2. *Left-right multiplications subgroup* $\mathcal{M}$: An $M \in \mathsf{GL}(n)$ is in $\mathcal{M}$ if and only if the full rank ABP $X_1 \cdot X_2 \cdots X_d$ determined by M has the following structure: There are matrices $A_1, \ldots, A_{d-1}$ with $A_k \in \mathsf{GL}(w_k)$ for every $k \in [d-1]$, such that $X_1 = Q_1 \cdot A_1$, $X_d = A_{d-1}^{-1} \cdot Q_d$, and $X_k = A_{k-1}^{-1} \cdot Q_k \cdot A_k$ for $k \in [2, d-1]$. It is easy to verify that $\mathcal{M}$ is a subgroup of $\mathcal{G}_{\mathsf{IMM}}$ and is isomorphic to the direct product $\mathsf{GL}(w_1) \times \ldots \times \mathsf{GL}(w_{d-1})$.
3. *Corner translations subgroup* $\mathcal{C}$: A $C \in \mathsf{GL}(n)$ is in $\mathcal{C}$ if and only if the full rank ABP $X_1 \cdot X_2 \cdots X_d$ determined by C has the following structure: There are two sets $\{C_{11}, C_{12}, \ldots, C_{1w_2}\}$ and $\{C_{d1}, C_{d2}, \ldots, C_{dw_{d-2}}\}$ containing anti-symmetric matrices in $\mathbb{F}^{w_1 \times w_1}$ and $\mathbb{F}^{w_{d-1} \times w_{d-1}}$ respectively such that $X_2 = Q_2 + Y_2$ and $X_{d-1} = Q_{d-1} + Y_{d-1}$, where $Y_2 \in \mathbb{F}[\mathbf{x}_1]^{w_1 \times w_2}$ (respectively $Y_{d-1} \in \mathbb{F}[\mathbf{x}_d]^{w_{d-2} \times w_{d-1}}$) is a matrix with its i-th column (respectively i-th row) equal to $C_{1i} \cdot Q_1^T$ (respectively $Q_d^T \cdot C_{di}$). For every other $k \in [d] \setminus \{2, d-1\}$, $X_k = Q_k$. Observe that $Q_1 \cdot C_{1i} \cdot Q_1^T = Q_d^T \cdot C_{di} \cdot Q_d = 0$. It can also be verified that $\mathcal{C}$ is an abelian subgroup of $\mathcal{G}_{\mathsf{IMM}}$ and is isomorphic to the direct product $\mathcal{A}_{w_1}^{w_2} \times \mathcal{A}_{w_{d-1}}^{w_{d-2}}$, where $\mathcal{A}_w$ is the group of $w \times w$ anti-symmetric matrices under matrix addition and $\mathcal{A}_w^k$ is the k times direct product of this group.

▶ **Theorem 54** (Symmetries of IMM). *$\mathcal{G}_{\mathsf{IMM}} = \mathcal{C} \rtimes \mathcal{H}$, where $\mathcal{H} = \mathcal{M} \rtimes \mathcal{T}$.*

We prove Theorem 54 below. Following are a couple of remarks on it.

Remarks

(a) *Characterization*: Let f be an n variate degree d polynomial satisfying the following: For any n variate degree d polynomial g, $\mathcal{G}_f = \mathcal{G}_g$ if and only if $f = \alpha \cdot g$ for some nonzero $\alpha \in \mathbb{F}$. Then f is said to be characterized by $\mathcal{G}_f$. We prove IMM is characterized by $\mathcal{G}_{\mathsf{IMM}}$ in Lemma 59. The groups $\mathcal{M}$ and $\mathcal{C}$ generate the 'continuous symmetries' of IMM.

(b) *Comparison with a related work*: In [15] a different choice of the IMM polynomial is considered, namely the trace of a product of d square symbolic matrices – let us call this polynomial IMM′[35]. The group of symmetries of IMM′ is determined in [15] and it is shown that IMM′ is characterized by $\mathcal{G}_{\mathsf{IMM}'}$. The group of symmetries of IMM′, like IMM, is generated by the transposition subgroup, the left-right multiplication subgroup, and (instead of the corner translations subgroup) the *circular transformations subgroup* – an element in this subgroup cyclically rotates the order of the matrices and hence does not change the trace of the product.

Proof of Theorem 54

We begin with the following observation which is immediate from Lemma 48.

▶ **Observation 55.** *If $X_1 \cdot X_2 \cdots X_d$ is a width $\mathbf{w}' = (w'_1, w'_2, \ldots, w'_{d-1})$ full rank ABP computing $\mathsf{IMM}_{\mathbf{w},d}$ then either*

1. *$w'_k = w_k$ for $k \in [d-1]$, and the spaces $\mathcal{X}_1, \mathcal{X}_{1,2}, \mathcal{X}_3, \ldots, \mathcal{X}_{d-1,d}, \mathcal{X}_d$ are the spaces $\mathcal{Q}_1, \mathcal{Q}_{1,2}, \mathcal{Q}_3, \ldots, \mathcal{Q}_{d-1,d}, \mathcal{Q}_d$ respectively, or*
2. *$w'_k = w_{d-k}$ for $k \in [d-1]$, and the spaces $\mathcal{X}_1, \mathcal{X}_{1,2}, \mathcal{X}_3, \ldots, \mathcal{X}_{d-1,d}, \mathcal{X}_d$ are the spaces $\mathcal{Q}_d, \mathcal{Q}_{d-1,d}, \mathcal{Q}_{d-2}, \ldots, \mathcal{Q}_{1,2}, \mathcal{Q}_1$ respectively.*

[35] The complexities of IMM and IMM′ are polynomially related to each other, in particular both are complete for algebraic branching programs under p-projections. But their groups of symmetries are slightly different.

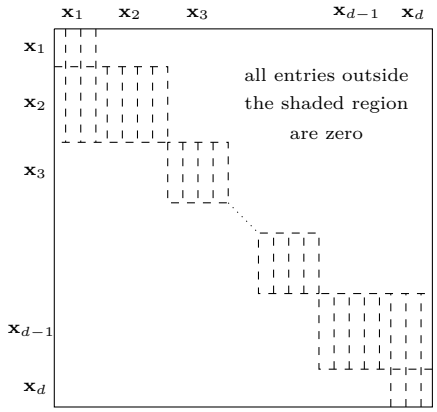


Figure 5 Matrix A in $\mathcal{G}_{\mathsf{IMM}}$.

From the definitions of $\mathcal{T}$, $\mathcal{M}$ and $\mathcal{C}$ it follows that $\mathcal{C} \cap \mathcal{M} = \mathcal{C} \cap \mathcal{T} = \mathcal{M} \cap \mathcal{T} = I_n$. The claim below shows $\mathcal{G}_{\mathsf{IMM}}$ is generated by $\mathcal{C}$, $\mathcal{M}$ and $\mathcal{T}$.

▶ **Claim 56.** *For every $A \in \mathcal{G}_{\mathsf{IMM}}$, there exist $C \in \mathcal{C}$, $M \in \mathcal{M}$ and $\tilde{T} \in \mathcal{T}$ such that $A = C \cdot M \cdot \tilde{T}$.*

Proof. Let $X_1 \cdot X_2 \cdots X_d$ be the full rank ABP $\mathtt{A}$ of width $\mathbf{w}$ determined by A. If $w_k = w_{d-k}$ for $k \in [d-1]$ then the spaces $\mathcal{X}_1, \mathcal{X}_{1,2}, \mathcal{X}_3, \ldots, \mathcal{X}_{d-1,d}, \mathcal{X}_d$ are either equal to $\mathcal{Q}_1, \mathcal{Q}_{1,2}, \mathcal{Q}_3, \ldots, \mathcal{Q}_{d-1,d}, \mathcal{Q}_d$ respectively or $\mathcal{Q}_d, \mathcal{Q}_{d-1,d}, \mathcal{Q}_{d-2}, \ldots, \mathcal{Q}_{1,2}, \mathcal{Q}_1$ respectively (from Observation 55). Otherwise if $w_k \neq w_{d-k}$ for any $k \in [d-1]$ then the spaces $\mathcal{X}_1, \mathcal{X}_{1,2}, \mathcal{X}_3, \ldots, \mathcal{X}_{d-1,d}, \mathcal{X}_d$ have only one choice and are equal to $\mathcal{Q}_1, \mathcal{Q}_{1,2}, \mathcal{Q}_3, \ldots, \mathcal{Q}_{d-1,d}, \mathcal{Q}_d$ respectively. We deal with these two choices of layer spaces separately.

Case A: Suppose $\mathcal{X}_1, \mathcal{X}_{1,2}, \mathcal{X}_3, \ldots, \mathcal{X}_{d-1,d}, \mathcal{X}_d$ are equal to $\mathcal{Q}_1, \mathcal{Q}_{1,2}, \mathcal{Q}_3, \ldots, \mathcal{Q}_{d-1,d}, \mathcal{Q}_d$ respectively. Hence A looks as shown in Figure 5.

The linear forms in X_2, X_{d-1} are in variables $\mathbf{x}_1 \uplus \mathbf{x}_2, \mathbf{x}_{d-1} \uplus \mathbf{x}_d$ respectively. Further,

$$\prod_{k=1}^{d} X_k = X_1 \cdot (X_2(\mathbf{x}_1) + X_2(\mathbf{x}_2)) \cdot \left(\prod_{k=3}^{d-2} X_k \right) \cdot (X_{d-1}(\mathbf{x}_{d-1}) + X_{d-1}(\mathbf{x}_d)) \cdot X_d = \mathsf{IMM}.^{36}$$

Since $\mathtt{A}$ is a full rank ABP and each monomial in IMM contains one variable from each set $\mathbf{x}_k$,

$$X_1 \cdot X_2(\mathbf{x}_2) \cdot \left(\prod_{k=3}^{d-2} X_k \right) \cdot X_{d-1}(\mathbf{x}_{d-1}) \cdot X_d = \mathsf{IMM}, \quad \text{and also}$$

$X_1 \cdot X_2(\mathbf{x}_1) \cdot \prod_{k=3}^{d-2} X_k \cdot X_{d-1}(\mathbf{x}_{d-1}) \cdot X_d = 0$ and $X_1 \cdot X_2(\mathbf{x}_2) \cdot \prod_{k=3}^{d-2} X_k \cdot X_{d-1}(\mathbf{x}_d) \cdot X_d = 0$ implying

$$X_1 \cdot X_2(\mathbf{x}_1) = 0_{w_2}^T \quad \text{and} \quad X_{d-1}(\mathbf{x}_d) \cdot X_d = 0_{w_{d-2}}, \tag{4}$$

where 0_w is a zero (column) vector in $\mathbb{F}^w$. Observation 57, the proof of which is in Section 7.5, proves the existence of a matrix $M \in \mathcal{M}$ such that the full rank ABP determined by M is $X_1 \cdot X_2(\mathbf{x}_2) \cdot X_3 \cdots X_{d-2} \cdot X_{d-1}(\mathbf{x}_{d-1}) \cdot X_d$.

▶ **Observation 57.** *There are matrices $A_1, \ldots, A_{d-1}$ with $A_k \in \mathsf{GL}(w_k)$ for every $k \in [d-1]$, such that $X_1 = Q_1 \cdot A_1$, $X_2(\mathbf{x}_2) = A_1^{-1} \cdot Q_2 \cdot A_2$, $X_{d-1}(\mathbf{x}_{d-1}) = A_{d-2}^{-1} \cdot Q_{d-1} \cdot A_{d-1}$, $X_d = A_{d-1}^{-1} \cdot Q_d$, and $X_k = A_{k-1}^{-1} \cdot Q_k \cdot A_k$ for $k \in [3, d-2]$.*

We now show the existence of a $C \in \mathcal{C}$ such that the full rank ABP determined by $C \cdot M$ is $X_1 \cdot X_2 \cdots X_d$, from which the claim follows by letting $\tilde{T} = I_n$. Since the linear forms in X_1 are $\mathbb{F}$-linearly independent, there are $w_1 \times w_1$ matrices $\{C_{11}, C_{12}, \ldots, C_{1w_2}\}$ such that the i-th column of $X_2(\mathbf{x}_1)$ is $C_{1i}X_1^T$. So from Equation (4), $X_1 \cdot C_{1i} \cdot X_1^T = 0$ (equivalently $Q_1 \cdot C_{1i} \cdot Q_1^T = 0$) implying C_{1i} is an anti-symmetric matrix for every $i \in [w_2]$. Similarly, there are $w_{d-1} \times w_{d-1}$ anti-symmetric matrices $\{C_{d1}, C_{d2}, \ldots, C_{dw_{d-2}}\}$ such that the i-th row of $X_{d-1}(\mathbf{x}_d)$ is $X_d^T C_{di}$. Let $C \in \mathsf{GL}(n)$ be such that the ABP determined by it is $Q_1 Q_2' Q_3 \cdots Q_{d-2} Q_{d-1}' Q_d$ where $Q_2' = Q_2 + Y_2$ and $Q_{d-1}' = Q_{d-1} + Y_{d-1}$, the i-th column (respectively i-th row) of Y_2 (respectively Y_{d-1}) is $C_{1i}Q_1^T$ (respectively $Q_{d-1}^T C_{di}$). By construction, $C \in \mathcal{C}$ and the ABP determined by $C \cdot M$ is $X_1 \cdot X_2 \cdots X_d$.

Case B: Suppose $\mathcal{X}_1, \mathcal{X}_{1,2}, \mathcal{X}_3, \ldots, \mathcal{X}_{d-1,d}, \mathcal{X}_d$ are the spaces $\mathcal{Q}_d, \mathcal{Q}_{d-1,d}, \mathcal{Q}_{d-2}, \ldots, \mathcal{Q}_{1,2}, \mathcal{Q}_1$ respectively. This implies $w_k = w_{d-k}$ for $k \in [d-1]$ and hence the full rank ABP determined by T is $Q_d^T \cdot Q_{d-1}^T \cdots Q_1^T$. From here the existence of $M \in \mathcal{M}$ and $C \in \mathcal{C}$ such that the full rank ABP determined by $M \cdot C \cdot T$ is $X_1 \cdot X_2 \cdots X_d$ follows just as in Case A. This completes the proof of the claim. ◀

Observe that if $T \in \mathcal{T}$ then for every $M \in \mathcal{M}$, $T \cdot M \cdot T^{-1} \in \mathcal{M}$. Let $\mathcal{H} = \mathcal{M} \rtimes \mathcal{T}$. Clearly, $\mathcal{C} \cap \mathcal{H} = I_n$. The following claim along with Claim 56 then conclude the proof of Theorem 54.

▶ **Claim 58.** *For every $C \in \mathcal{C}$ and $H \in \mathcal{H}$, $H \cdot C \cdot H^{-1} \in \mathcal{C}$.*

Proof. Let $H = M \cdot T$ where $M \in \mathcal{M}$ and $T \in \mathcal{T}$, and $A = MT \cdot C \cdot T^{-1}M^{-1}$. Suppose $X_1 \cdot X_2 \cdots X_{d-1} \cdot X_d$ is the ABP determined by A. The matrices T and T^{-1} in A together ensure that the spaces $\mathcal{X}_1, \mathcal{X}_{1,2}, \mathcal{X}_3, \ldots, \mathcal{X}_{d-1,d}, \mathcal{X}_d$ are equal to $\mathcal{Q}_1, \mathcal{Q}_{1,2}, \mathcal{Q}_3, \ldots, \mathcal{Q}_{d-1,d}, \mathcal{Q}_d$ respectively. Also the matrices M and M^{-1} together ensure that $X_i = Q_i$ for $i \in [d] \setminus \{2, d-1\}$, $X_2(\mathbf{x}_2) = Q_2$ and $X_{d-1}(\mathbf{x}_{d-1}) = Q_{d-1}$. Arguing as in Claim 56, we can infer that $A \in \mathcal{C}$. ◀

6.2 Characterization of IMM by $\mathcal{G}_{\mathsf{IMM}}0$

For every $f = \alpha \cdot \mathsf{IMM}$, where $\alpha \in \mathbb{F}$ and $\alpha \neq 0$, it is easily observed that $\mathcal{G}_f = \mathcal{G}_{\mathsf{IMM}}$. We prove the converse in the following lemma for any homogeneous degree d polynomial in the $\mathbf{x}$ variables.

▶ **Lemma 59.** *Let f be a homogeneous degree d polynomial in n variables $\mathbf{x} = \mathbf{x}_1 \uplus \ldots \uplus \mathbf{x}_d$. If $|\mathbb{F}| > d+1$ and the left-right multiplications subgroup $\mathcal{M}$ of $\mathcal{G}_{\mathsf{IMM}}$ is contained in $\mathcal{G}_f$ then $f = \alpha \cdot \mathsf{IMM}$ for some nonzero $\alpha \in \mathbb{F}$.*

Proof. First, we show that such an f is set-multilinear in the sets $\mathbf{x}_1, \ldots, \mathbf{x}_d$: Every monomial in f has exactly one variable from each of the sets $\mathbf{x}_1, \ldots, \mathbf{x}_d$. As $|\mathbb{F}| > d+1$, there is a nonzero $\rho \in \mathbb{F}$ that is not an e-th root of unity for any $e \leq d$. Every element in $\mathcal{M}$ is defined by $d-1$ matrices $A_1, \ldots, A_{d-1}$ such that $A_k \in \mathsf{GL}(w_k)$ for every $k \in [d-1]$. For a $k \in [d-1]$, consider the element $M \in \mathcal{M}$ that is defined by $A_k = \rho \cdot I_{w_k}$ and $A_l = I_{w_l}$ for $l \in [d-1]$ and $l \neq k$. Then, $f(M \cdot \mathbf{x}) = f(\mathbf{x}_1, \ldots, \rho\mathbf{x}_k, \rho^{-1}\mathbf{x}_{k+1}, \ldots, \mathbf{x}_d)$, which by assumption is f. Comparing the coefficients of the monomials of $f(M \cdot \mathbf{x})$ and f, we observe that in every monomial of f the number of variables from $\mathbf{x}_k$ and $\mathbf{x}_{k+1}$ must be the same as ρ is not an e-th root of unity for any $e \leq d$. Since k is chosen arbitrarily and f is homogeneous of degree d, f must be set-multilinear in the sets $\mathbf{x}_1, \ldots, \mathbf{x}_d$.

The proof is by induction on the degree of f. For $i \in [w_1]$, let $\mathbf{x}_{2i}$ be the set of variables in the i-th row of Q_2, and Q_{2i} be the $1 \times w_2$ matrix containing the i-th row of Q_2. Let IMM_i be the degree $d-1$ iterated matrix multiplication polynomial computed by the ABP $Q_{2i} \cdot Q_3 \cdots Q_d$. As f is set-multilinear, it can be expressed as

$$f = g_1 \cdot x_1^{(1)} + \ldots + g_{w_1} \cdot x_{w_1}^{(1)}, \tag{5}$$

where $g_1, \ldots, g_{w_1}$ are set-multilinear polynomials in the sets $\mathbf{x}_2, \ldots, \mathbf{x}_d$. Moreover, we can argue that g_i is set-multilinear in $\mathbf{x}_{2i}, \mathbf{x}_3, \ldots, \mathbf{x}_d$ as follows: Consider an $N \in \mathcal{M}$ that is defined by a diagonal matrix $A_1 \in \mathsf{GL}(w_1)$ whose (i,i)-th entry is ρ and all other diagonal entries are 1; every other $A_l = I_{w_l}$ for $l \in [2, d-1]$. The transformation N scales the variable $x_i^{(1)}$ by ρ and the variables in $\mathbf{x}_{2i}$ by ρ^{-1}. By comparing the coefficients of the monomials of $f(N \cdot \mathbf{x})$ and f, we can conclude that g_i is set-multilinear in $\mathbf{x}_{2i}, \mathbf{x}_3, \ldots, \mathbf{x}_d$ for every $i \in [w_1]$.

Let $\mathcal{M}'$ be the subgroup of $\mathcal{M}$ containing those $M \in \mathcal{M}$ for which $A_1 = I_{w_1}$. From Equation (5), we can infer that $g_i(M \cdot \mathbf{x}) = g_i$ for $M \in \mathcal{M}'$, and hence the left-right multiplications subgroup of $\mathcal{G}_{\mathsf{IMM}_i}$ is contained in the group of symmetries of g_i. As degree of g_i is $d-1$, by induction[37] $g_i = \alpha_i \cdot \mathsf{IMM}_i$ for some nonzero $\alpha_i \in \mathbb{F}$ and

$$f = \alpha_1 \cdot \mathsf{IMM}_1 \cdot x_1^{(1)} + \ldots + \alpha_{w_1} \cdot \mathsf{IMM}_{w_1} \cdot x_{w_1}^{(1)}. \tag{6}$$

Next we show that $\alpha_1 = \ldots = \alpha_{w_1}$ thereby completing the proof.

For an $i \in [2, w_1]$, let $A_1 \in \mathsf{GL}(w_1)$ be the upper triangular matrix whose diagonal entries are 1, the $(1,i)$-th entry is 1 and remaining entries are zero. Let U be the matrix in $\mathcal{M}$ defined by A_1 and $A_l = I_{w_l}$ for $l \in [2, d-1]$. The transformation U has the following effect on the variables:

$$\begin{aligned} x_i^{(1)} &\mapsto x_1^{(1)} + x_i^{(1)} \quad \text{and} \\ x_{1j}^{(2)} &\mapsto x_{1j}^{(2)} - x_{ij}^{(2)} \quad \text{for every } j \in [w_2], \end{aligned}$$

every other $\mathbf{x}$ variable maps to itself. Applying U to f in Equation (6) we get

$$\begin{aligned} f &= f(U \cdot \mathbf{x}) = \alpha_1 \cdot (\mathsf{IMM}_1 - \mathsf{IMM}_i) \cdot x_1^{(1)} + \ldots + \alpha_i \cdot \mathsf{IMM}_i \cdot (x_1^{(1)} + x_i^{(1)}) + \ldots + \\ &\quad \alpha_{w_1} \cdot \mathsf{IMM}_{w_1} \cdot x_{w_1}^{(1)} \\ &= f + (\alpha_i - \alpha_1) \cdot \mathsf{IMM}_i \cdot x_1^{(1)}, \\ &\Rightarrow \alpha_i - \alpha_1 = 0. \end{aligned}$$

Since this is true for any $i \in [2, w_1]$, we have $\alpha_1 = \ldots = \alpha_{w_1}$. ◀

7 Proof of claims and lemmas from previous sections

In this section we give proofs of claims and lemmas from the above sections. We begin by proving the incompleteness of the full rank ABP.

▶ **Observation 60.** *For every sufficiently large $m \in \mathbb{N}$ there is an m variate multilinear polynomial that is not computable by full rank ABP.*

Proof. A full rank ABP computing an m variate polynomial f has both its width and length bounded by m, so f can also be computed by an ABP (not full rank) of width and length exactly m. Hence, it is sufficient to show that there is an m variate multilinear polynomial

[37] The base case $d = 1$ is trivial to show.

that is not computable by the latter kind of ABP. The number of edges in an ABP of width m and length m is $n = m^2(m-2) + 2m$. Let these n edges be $e_1, e_2, \ldots, e_n$ and suppose the edge e_i is labelled by the affine form $l_i = \sum_{j=1}^{m} c_{ij}x_j + c_{i0}$. Treat c_{ij}'s as formal variables. Then each of the $\binom{2n}{n}$ coefficients of the polynomial f computed by such an ABP is a polynomial in these $n(m+1)$ formal variables. Since $n(m+1) < 2^m$ for sufficiently large m, the coefficients of f restricted to just the multilinear monomials $\mathfrak{m}_1, \mathfrak{m}_2, \ldots, \mathfrak{m}_{2^m}$ are algebraically dependent. Let $h \neq 0$ be an annihilating polynomial of these coefficients. Since h is nonzero, there is a point $\mathbf{a} = (a_1, \ldots, a_{2^m}) \in \mathbb{F}^{2^m}$ such that $h(\mathbf{a}) \neq 0$. It follows that the multilinear polynomial $g \stackrel{\text{def}}{=} \sum_{i=1}^{2^m} a_i\mathfrak{m}_i$ is not computable by an ABP of width m and length m, which means g is not computable by a full rank ABP. ◀

7.1 Proof of lemmas and claims in Section 2

▶ **Claim 21** (restated). *If $f(\mathbf{x}) = g(A\mathbf{x})$, where f and g are both n variate polynomials and $A \in \mathsf{GL}(n)$, then the Lie algebra of f is a conjugate of the Lie algebra of g via A, i.e. $\mathfrak{g}_f = \{A^{-1}EA \ : \ E \in \mathfrak{g}_g\} =: A^{-1}\mathfrak{g}_g A$.*

Proof. Let $Q = (q_{i,j})_{i,j\in[n]} \in \mathfrak{g}_f$. Hence,

$$\sum_{i,j\in[n]} q_{ij}x_j \cdot \frac{\partial f}{\partial x_i} = 0 \quad \Rightarrow \quad \sum_{i,j\in[n]} q_{ij}x_j \cdot \frac{\partial g(A\mathbf{x})}{\partial x_i} = 0 \ . \tag{7}$$

Let $A = (a_{ki})_{k,i\in[n]}$. Using chain rule of derivatives,

$$\frac{\partial g(A\mathbf{x})}{\partial x_i} = \sum_{k\in[n]} \frac{\partial g}{\partial x_k}(A\mathbf{x}) \cdot a_{ki} \quad .$$

Let $A^{-1} = (b_{jl})_{j,l\in[n]}$ and $(A\mathbf{x})_l$ be the l-th entry of $A\mathbf{x}$. Then $x_j = \sum_{l\in[n]} b_{jl}(A\mathbf{x})_l$. From Equation (7),

$$\begin{aligned}
\sum_{i,j\in[n]} q_{ij} \cdot \left(\sum_{l\in[n]} b_{jl}(A\mathbf{x})_l\right) \cdot \left(\sum_{k\in[n]} \frac{\partial g}{\partial x_k}(A\mathbf{x}) \cdot a_{ki}\right) &= 0 \ , \\
\Rightarrow \sum_{k,l\in[n]} (A\mathbf{x})_l \cdot \frac{\partial g}{\partial x_k}(A\mathbf{x}) \cdot \left(\sum_{i,j\in[n]} a_{ki}q_{ij}b_{jl}\right) &= 0 \ , \\
\Rightarrow \sum_{k,l\in[n]} x_l \cdot \frac{\partial g}{\partial x_k} \cdot \left(\sum_{i,j\in[n]} a_{ki}q_{ij}b_{jl}\right) &= 0 \quad \text{(Substituting } \mathbf{x} \text{ by } A^{-1}\mathbf{x}\text{)}.
\end{aligned}$$

Observe that $\sum_{i,j\in[n]} a_{ki}q_{ij}b_{jl}$ is the (k,l)-th entry of AQA^{-1}. Hence, $AQA^{-1} \in \mathfrak{g}_g$ implying $\mathfrak{g}_f \subseteq A^{-1}\mathfrak{g}_g A$. Similarly, $\mathfrak{g}_g \subseteq A\mathfrak{g}_f A^{-1}$ as $g = f(A^{-1}\mathbf{x})$, implying $\mathfrak{g}_f = A^{-1}\mathfrak{g}_g A$. ◀

▶ **Claim 24** (restated). *With probability at least $1 - \frac{1}{\text{poly}(n)}$, the rank of the matrix $M = (f_j(\mathbf{b}_i))$ where $i, j \in [m]$, is $m - r$ where $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_m$ are chosen independently and uniformly at random from $S^n \subset \mathbb{F}^n$ with $|S| = dm \cdot \text{poly}(n)$.*

Proof. Recall, we assumed that the dimension of the $\mathbb{F}$-linear space spanned by the n variate polynomials $f_1, f_2, \ldots, f_m$ is $m - r$. Without loss of generality assume $f_1, f_2, \ldots, f_{m-r}$ form a basis of this linear space. Clearly, the rank of $M = (f_j(\mathbf{b}_i))_{i,j\in[m]}$ is less than or equal to $m-r$. Let $M_{m-r} = (f_j(\mathbf{b}_i))_{i,j\in[m-r]}$. That $\mathsf{Det}(M_{m-r}) \neq 0$ with probability at least $1 - \frac{1}{\text{poly}(n)}$ over

the random choices of $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_m$ can be argued as follows: Let $\mathbf{y}_i = \{y_1^{(i)}, y_2^{(i)}, \ldots, y_n^{(i)}\}$ for $i \in [m-r]$ be disjoint sets of variables. Rename the $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$ variables in $f_j(\mathbf{x})$ to $\mathbf{y}_i$ and call these new polynomials $f_j(\mathbf{y}_i)$ for $i, j \in [m-r]$. Let Y be an $(m-r) \times (m-r)$ matrix whose (i,j)-th entry is $(f_j(\mathbf{y}_i))_{i \in [m-r]}$. Since $f_1, f_2, \ldots, f_{m-r}$ are $\mathbb{F}$-linearly independent, $\mathsf{Det}(Y) \neq 0$ – this can be argued easily using induction. As $\deg(\mathsf{Det}(Y)) = d(m-r) \leq dm$, by Schwartz-Zippel lemma, $\mathsf{Det}(M_{m-r}) \neq 0$ with probability at least $1 - \frac{1}{\mathrm{poly}(n)}$. ◀

▶ **Claim 26** (restated). *Let r be the number of redundant variables in an n variate polynomial f of degree d. Then the dimension of the space $\mathcal{U}$ of $\mathbb{F}$-linear dependencies of $\{\partial_{x_i} f \mid i \in [n]\}$ is r. Moreover, we can construct an $A \in \mathsf{GL}(n)$ in randomized $\mathrm{poly}(n, d, \beta)$ time such that $f(A\mathbf{x})$ is free of the set of variables $\{x_{n-r+1}, x_{n-r+2}, \ldots, x_n\}$ with high probability, where β is the bit length of the coefficients of f.*

Proof. Let $B = (b_{ij})_{i,j \in [n]} \in \mathsf{GL}(n)$ such that $f(B\mathbf{x})$ is a polynomial in $x_1, x_2, \ldots, x_s$, where $s = n - r$. For $n - r + 1 \leq j \leq n$

$$\begin{aligned} & \frac{\partial f(B\mathbf{x})}{\partial x_j} = 0 \\ \Rightarrow \quad & \sum_{i=1}^{n} b_{ij} \cdot \frac{\partial f}{\partial x_i}(B\mathbf{x}) = 0 \quad \text{(by chain rule)} \\ \Rightarrow \quad & \sum_{i=1}^{n} b_{ij} \cdot \frac{\partial f}{\partial x_i} = 0 \quad \text{(substituting } \mathbf{x} \text{ by } B^{-1}\mathbf{x}\text{)}. \end{aligned}$$

Since $B \in \mathsf{GL}(n)$, we conclude $\dim(\mathcal{U}) \geq r$. Let $\{(a_{1j}\ a_{2j} \ldots\ a_{nj})^T : (n - \dim(\mathcal{U}) + 1) \leq j \leq n\}$ be a basis of $\mathcal{U}$. Then,

$$\sum_{i=1}^{n} a_{ij} \cdot \frac{\partial f}{\partial x_i} = 0.$$

Let $A \in \mathsf{GL}(n)$ such that for $(n - \dim(\mathcal{U}) + 1) \leq j \leq n$, the j-th column of A is $(a_{1j}\ a_{2j} \ldots\ a_{nj})^T$ and the remaining columns of A are arbitrary vectors that make A a full rank matrix. Then,

$$\sum_{i=1}^{n} a_{ij} \cdot \frac{\partial f}{\partial x_i} = 0 \quad \Rightarrow \quad \sum_{i=1}^{n} a_{ij} \cdot \frac{\partial f}{\partial x_i}(A\mathbf{x}) = 0 \quad \Rightarrow \quad \frac{\partial f(A\mathbf{x})}{\partial x_j} = 0.$$

This implies $f(A\mathbf{x})$ is a polynomial free of x_j variable for $(n - \dim(\mathcal{U}) + 1) \leq j \leq n$. Hence, $\dim(\mathcal{U}) \leq r$.

Blackbox for polynomials $\partial_{x_1} f, \partial_{x_2} f, \ldots, \partial_{x_n} f$ can be constructed in $\mathrm{poly}(n, d, \beta)$ time from blackbox access to f and a basis for the space $\mathcal{U}$ of $\mathbb{F}$-linear dependencies of polynomials $\partial_{x_1} f, \partial_{x_2} f, \ldots, \partial_{x_n} f$ can also be constructed in randomized $\mathrm{poly}(n, d, \beta)$ time (see Section 2.2). Thus, we can construct an $A \in \mathsf{GL}(n)$ (similar to the construction shown above) from a blackbox access to f in randomized $\mathrm{poly}(n, d, \beta)$ time such that $f(A\mathbf{x})$ is free of the set of variables $\{x_{n-r+1}, x_{n-r+2}, \ldots, x_n\}$. We summarize this in Algorithm 8. ◀

▶ **Lemma 27** (restated). *There is a randomized algorithm that takes input blackbox access to two n variate, degree d polynomials f and g, and with probability at least $1 - \frac{1}{\mathrm{poly}(n)}$ does the following: if f is translation equivalent to g, outputs an $\mathbf{a} \in \mathbb{F}^n$ such that $f(\mathbf{x} + \mathbf{a}) = g(\mathbf{x})$, else outputs 'f and g are not translation equivalent'. The running time of the algorithm is $\mathrm{poly}(n, d, \beta)$, where β is the bit length of the coefficients of f and g.*

Algorithm 8 Eliminating redundant variables

INPUT: Blackbox access to an n variate polynomial $f(\mathbf{x})$.

OUTPUT: An r and an $A \in \mathsf{GL}(n)$ such that r is the number of redundant variables in f and $f(A\mathbf{x})$ is free of the variables $x_{n-r+1}, x_{n-r+2}, \ldots, x_n$.

1. Compute blackbox access to $\partial_{x_1} f, \partial_{x_2} f, \ldots, \partial_{x_n} f$ (see Section 2.2).
2. Compute a basis $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_r\}$ of the space of $\mathbb{F}$-linear dependencies of $\partial_{x_1} f, \partial_{x_2} f, \ldots, \partial_{x_n} f$ (using the random substitution idea in Claim 24). /* This step succeeds in computing the required basis with high probability. */
3. Construct an $A \in \mathsf{GL}(n)$ such that the last r columns of A are $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_r$ and the remaining columns of A are chosen arbitrarily to make A a full rank matrix.
4. Return r and A.

Proof. We present the algorithm formally in Algorithm 9. If it succeeds in computing a point $\mathbf{a} \in \mathbb{F}^n$ in the end (in step 20), it performs a randomized blackbox polynomial identity test (PIT) to check whether $f(\mathbf{x}+\mathbf{a}) = g(\mathbf{x})$ (in step 22). If f and g are not translation equivalent, this final PIT finds it with probability at least $1 - \frac{1}{\text{poly}(n)}$. So, for the analysis of the algorithm we can assume there is an $\mathbf{a} = (a_1\ a_2\ \ldots\ a_n)^T \in \mathbb{F}^n$ such that $f(\mathbf{x}+\mathbf{a}) = g(\mathbf{x})$. The strategy outlined below helps to argue the correctness of Algorithm 9.

Strategy: Suppose $f(\mathbf{x}+\mathbf{a}) = g(\mathbf{x})$. By equating the degree d and degree $d-1$ homogeneous components of f and g we get the following equations,

$$f^{[d]} = g^{[d]} \quad \text{and}$$
$$f^{[d-1]} + \sum_{i=1}^{n} a_i \cdot \frac{\partial f^{[d]}}{\partial x_i} = g^{[d-1]} \quad \Rightarrow \quad \sum_{i=1}^{n} a_i \cdot \frac{\partial f^{[d]}}{\partial x_i} = g^{[d-1]} - f^{[d-1]}. \tag{8}$$

Let $f_i = \frac{\partial f^{[d]}}{\partial x_i}$ for $i \in [n]$. Blackbox access to the homogeneous components of f: $f^{[0]}, f^{[1]}, \ldots, f^{[d]}$, the homogeneous components of g: $g^{[0]}, g^{[1]}, \ldots, g^{[d]}$ and $f_1, f_2, \ldots f_n$ can be constructed from blackbox access to f and g in $\text{poly}(n, d, \beta)$ time (see Section 2.2). If $f_1, f_2, \ldots, f_n$ are $\mathbb{F}$-linearly independent then with high probability over the random choices of $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n \in \mathbb{F}^n$ the matrix $(f_j(\mathbf{b}_i))_{i,j\in[n]}$ has full rank (from Claim 24). Hence, we can solve for $a_1, a_2, \ldots, a_n$ uniquely from Equation (8). In the general case (when $f_1, f_2, \ldots, f_n$ may be $\mathbb{F}$-linearly dependent), the algorithm repeatedly applies *variable reduction* and *degree reduction* (as described below) to compute $\mathbf{a}$.

Variable reduction. We construct a transformation $A \in \mathsf{GL}(n)$ such that $f^{[d]}(A\mathbf{x})$ has only the essential variables $x_1, \ldots, x_m$ (see Claim 26). Let $\tilde{f} = f(A\mathbf{x})$, $\tilde{g} = g(A\mathbf{x})$. It is sufficient to compute a point $\mathbf{b} = (b_1\ b_2\ \ldots\ b_n)^T \in \mathbb{F}^n$ such that $\tilde{f}(\mathbf{x}+\mathbf{b}) = \tilde{g}(\mathbf{x})$ as

$$\tilde{f}(\mathbf{x}+\mathbf{b}) = \tilde{g}(\mathbf{x}) \quad \Rightarrow \quad f(A\mathbf{x}+A\mathbf{b}) = g(A\mathbf{x}) \quad \Rightarrow \quad f(\mathbf{x}+A\mathbf{b}) = g(\mathbf{x}).$$

So we can choose $\mathbf{a} = A\mathbf{b}$. As in Equation (8),

$$\tilde{f}^{[d]} = \tilde{g}^{[d]} \qquad \text{and} \qquad \sum_{i=1}^{m} b_i \cdot \frac{\partial \tilde{f}^{[d]}}{\partial x_i} = \tilde{g}^{[d-1]} - \tilde{f}^{[d-1]}. \tag{9}$$

The derivatives $\partial_{x_i} \tilde{f}^{[d]}$ for $i > m$ are zero as $\tilde{f}^{[d]} = f^{[d]}(A\mathbf{x})$ has only the essential variables $x_1, x_2, \ldots, x_m$. Also the polynomials $\{\partial_{x_i} \tilde{f}^{[d]} \ : \ i \in [m]\}$ are $\mathbb{F}$-linearly independent (by Claim 26). Hence, we can solve for unique $b_1, b_2, \ldots, b_m$ satisfying Equation (9) as before.

Degree reduction. To compute $b_{m+1}, b_{m+2}, \ldots, b_n$ we reduce the problem to finding a point that asserts translation equivalence of two degree $d-1$ polynomials. Let $\mathbf{b'} = (b_1\ b_2\ \ldots\ b_m\ 0\ \ldots\ 0)^T$, $\widehat{f} = \tilde{f}(\mathbf{x}+\mathbf{b'})$. Further, let $\mathbf{e} \in \mathbb{F}^n$ such that $\widehat{f}(\mathbf{x}+\mathbf{e}) = \tilde{g}(\mathbf{x})$. Then the first m coordinates of $\mathbf{e}$ must be zero[38] and we can choose $\mathbf{b} = \mathbf{b'} + \mathbf{e}$. We have the following equations,

$$\widehat{f}^{[d]}(\mathbf{x}+\mathbf{e}) + (\widehat{f} - \widehat{f}^{[d]})(\mathbf{x}+\mathbf{e}) = \tilde{g}^{[d]}(\mathbf{x}) + (\tilde{g} - \tilde{g}^{[d]})(\mathbf{x})$$

$$\Leftrightarrow \quad \tilde{f}^{[d]}(\mathbf{x}+\mathbf{e}) + (\widehat{f} - \widehat{f}^{[d]})(\mathbf{x}+\mathbf{e}) = \tilde{g}^{[d]}(\mathbf{x}) + (\tilde{g} - \tilde{g}^{[d]})(\mathbf{x}) \quad (\text{as } \widehat{f}^{[d]} = \tilde{f}^{[d]}).$$

Since $\tilde{f}^{[d]}$ has only $x_1, x_2, \ldots, x_m$ variables and the first m coordinates of $\mathbf{e}$ are zero, the above statement is equivalent to

$$\tilde{f}^{[d]}(\mathbf{x}) + (\widehat{f} - \widehat{f}^{[d]})(\mathbf{x}+\mathbf{e}) = \tilde{g}^{[d]}(\mathbf{x}) + (\tilde{g} - \tilde{g}^{[d]})(\mathbf{x})$$

$$\Leftrightarrow \quad (\widehat{f} - \widehat{f}^{[d]})(\mathbf{x}+\mathbf{e}) = (\tilde{g} - \tilde{g}^{[d]})(\mathbf{x}) \quad (\text{from Equation (9)}).$$

The polynomials $\widehat{f} - \widehat{f}^{[d]}$ and $\tilde{g} - \tilde{g}^{[d]}$ have degree at most $d-1$ and blackboxes for these polynomials can be constructed in $\text{poly}(n, d, \beta)$ time. Therefore the problem reduces to computing a point $\mathbf{e} \in \mathbb{F}^n$ that asserts translation equivalence of two degree $(d-1)$ polynomials.

Correctness of Algorithm 9: In steps 4–11, the algorithm carries out variable reduction and computes a part of the translation $\mathbf{b}$ that we call $\mathbf{b'}$ in the above argument. The remaining part of $\mathbf{b}$ (which is the vector $\mathbf{e}$ above) is computed by carrying out degree reduction in step 12 and then inducting on lower degree polynomials. These parts are then added appropriately in step 17, and finally an $\mathbf{a}$ is recovered in step 20. ◀

▶ **Lemma 28** (restated). *There is a randomized algorithm which when given blackbox access to an n variate degree d polynomial f, computes a basis of $\mathfrak{g}_f$ with probability at least $1 - \frac{1}{\text{poly}(n)}$ in time $\text{poly}(n, d, \beta)$ where β is the bit length of the coefficients in f.*

Proof. Recall, the Lie algebra of f is the set of all matrices $E = (e_{ij})_{i,j\in[n]}$ such that $\sum_{i,j\in[n]} e_{ij} x_j \cdot \frac{\partial f}{\partial x_i} = 0$. Hence, $\mathfrak{g}_f$ is the space of linear dependencies of the polynomials $x_j \cdot \frac{\partial f}{\partial x_i}$ for $i, j \in [n]$. Using Claim 23, we can derive blackboxes for these n^2 polynomials and then compute a basis of the space of linear dependencies with high probability using Claim 24. ◀

7.2 Proof of lemmas and claims in Section 3

▶ **Lemma 31** (restated). *Let $\mathcal{W}_1, \mathcal{W}_2, \mathcal{W}_3$ be the following sets (spaces) of matrices:*

1. *$\mathcal{W}_1$ consists of all matrices $D = (d_{ij})_{i,j\in[n]}$ such that D is diagonal and*

$$\sum_{i=1}^{n} d_{ii} x_i \cdot \frac{\partial \mathsf{IMM}}{\partial x_i} = 0.$$

2. *$\mathcal{W}_2$ consists of all matrices $B = (b_{ij})_{i,j\in[n]}$ such that*

$$\sum_{i,j\in[n]} b_{ij} x_j \cdot \frac{\partial \mathsf{IMM}}{\partial x_i} = 0,$$

where in every summand $b_{ij} \neq 0$ only if $x_i \neq x_j$ and $x_i, x_j \in \mathbf{x}_l$ for some $l \in [d]$.

[38] As $b_1, b_2, \ldots, b_m$ can be solved uniquely.

Algorithm 9 Translation equivalence test

INPUT: Blackbox access to two n variate, degree d polynomials f and g.

OUTPUT: A point $\mathbf{a} \in \mathbb{F}^n$ such that $f(\mathbf{x}+\mathbf{a}) = g(\mathbf{x})$, if such an $\mathbf{a}$ exists.

1. Set $\ell = d$, $p = f$ and $q = g$.
2.
3. **while** $\ell > 0$ **do**
4. Using Algorithm 8 find an m and an $A_\ell \in \mathsf{GL}(n)$ such that the variables $x_{m+1}, x_{m+2}, \ldots, x_n$ do not appear in $p^{[\ell]}(A_\ell \mathbf{x})$. /* With high probability m is the number of essential variables in $p^{[\ell]}$. */
5. Let $\tilde{p} = p(A_\ell \mathbf{x})$ and $\tilde{q} = q(A_\ell \mathbf{x})$. Construct blackbox access to $\tilde{p}^{[\ell]}, \tilde{p}^{[\ell-1]}, \tilde{q}^{[\ell]}, \tilde{q}^{[\ell-1]}$ and $\partial_{x_i} \tilde{p}^{[\ell]}$ for $i \in [m]$.
6. Check if $\tilde{p}^{[\ell]} = \tilde{q}^{[\ell]}$. If not, output '$f$ and g are not translation equivalent' and stop. /* The check succeeds with high probability. */
7. Solve for unique $b_1, b_2, \ldots, b_m$ satisfying

$$\sum_{i=1}^{m} b_i \cdot \frac{\partial \tilde{p}^{[\ell]}}{\partial x_i} = \tilde{q}^{[\ell-1]} - \tilde{p}^{[\ell-1]} \quad \text{(using the random substitution idea in Claim 24).}$$

If the solving fails, output 'f and g are not translation equivalent'. /* This step succeeds with high probability if m is the number of essential variables in $p^{[\ell]}$ in step 4. */
8. **if** $m = n$ **then**
9. Set $\mathbf{b}_\ell = (b_1\ b_2\ \ldots\ b_n)$ and exit while loop.
10. **else**
11. Set $\mathbf{b}_\ell = (b_1\ b_2\ \ldots b_m\ 0\ \ldots\ 0) \in \mathbb{F}^n$.
12. Construct blackbox access to $(\tilde{p} - \tilde{p}^{[\ell]})(\mathbf{x} + \mathbf{b}_\ell)$ and $(\tilde{q} - \tilde{q}^{[\ell]})(\mathbf{x})$. Set $p = (\tilde{p} - \tilde{p}^{[\ell]})(\mathbf{x} + \mathbf{b}_\ell)$, $q = (\tilde{q} - \tilde{q}^{[\ell]})(\mathbf{x})$ and $\ell = \ell - 1$.
13. **end if**
14. **end while**
15.
16. **while** $\ell < d$ **do**
17. Set $\mathbf{b}_{\ell+1} = \mathbf{b}_{\ell+1} + A_\ell \mathbf{b}_\ell$.
18. Set $\ell = \ell + 1$.
19. **end while**
20. Set $\mathbf{a} = A_d \mathbf{b}_d$.
21.
22. Pick a point $\mathbf{c}$ uniformly at random from $S^n \subset \mathbb{F}^n$ with $|S| = d.\mathrm{poly}(n)$ and check whether $f(\mathbf{c}+\mathbf{a}) = g(\mathbf{c})$. /* With high probability $f(\mathbf{c}+\mathbf{a}) \neq g(\mathbf{c})$ if f and g are not translation equivalent.*/
23. **if** $f(\mathbf{c}+\mathbf{a}) = g(\mathbf{c})$ **then**
24. Output the point $\mathbf{a}$.
25. **else**
26. Output 'f and g are not translation equivalent'.
27. **end if**

3. $\mathcal{W}_3$ *consists of all matrices* $C = (c_{ij})_{i,j\in[n]}$ *such that*

$$\sum_{i,j\in[n]} c_{ij}x_j \cdot \frac{\partial \mathsf{IMM}}{\partial x_i} = 0,$$

where in every summand $c_{ij} \neq 0$ *only if either* $x_i \in \mathbf{x}_2$, $x_j \in \mathbf{x}_1$ *or* $x_i \in \mathbf{x}_{d-1}$, $x_j \in \mathbf{x}_d$. *Then* $\mathfrak{g}_{\mathsf{IMM}} = \mathcal{W}_1 \oplus \mathcal{W}_2 \oplus \mathcal{W}_3$.

Proof. Since $\mathcal{W}_1 \cap \mathcal{W}_2 = (\mathcal{W}_1 + \mathcal{W}_2) \cap \mathcal{W}_3 = \{\mathbf{0}_n\}$, where $\mathbf{0}_n$ is the $n \times n$ all zero matrix, it is sufficient to show $\mathfrak{g}_{\mathsf{IMM}} = \mathcal{W}_1 + \mathcal{W}_2 + \mathcal{W}_3$. By definition, $\mathcal{W}_1 + \mathcal{W}_2 + \mathcal{W}_3 \subseteq \mathfrak{g}_{\mathsf{IMM}}$. We now show that $\mathfrak{g}_{\mathsf{IMM}} \subseteq \mathcal{W}_1 + \mathcal{W}_2 + \mathcal{W}_3$. Let $E = (e_{ij})_{i,j\in[n]}$ be a matrix in $\mathfrak{g}_{\mathsf{IMM}}$. Then $\sum_{i,j\in[n]} e_{ij}x_j \cdot \frac{\partial \mathsf{IMM}}{\partial x_i} = 0$. We focus on a term $x_j \cdot \frac{\partial \mathsf{IMM}}{\partial x_i}$ and observe the following:

(a) If $x_i = x_j$ then the monomials of $x_i \cdot \frac{\partial \mathsf{IMM}}{\partial x_i}$ are also monomials of IMM. Such monomials do not appear in any term $x_j \cdot \frac{\partial \mathsf{IMM}}{\partial x_i}$, where $x_i \neq x_j$.

(b) If $x_i \neq x_j$ and x_i, x_j belong to the same $\mathbf{x}_l$ then every monomial in $x_j \cdot \frac{\partial \mathsf{IMM}}{\partial x_i}$ has exactly one variable from every $\mathbf{x}_k$ for $k \in [d]$. Such monomials do not appear in a term $x_j \cdot \frac{\partial \mathsf{IMM}}{\partial x_i}$, where $x_i \in \mathbf{x}_l$ and $x_j \in \mathbf{x}_k$ and $l \neq k$.

Due to this monomial disjointness, an equation $\sum_{i,j\in[n]} e_{ij}x_j \cdot \frac{\partial \mathsf{IMM}}{\partial x_i} = 0$ corresponding to E can be split into three equations:

1. $\sum_{i=1}^{n} d_{ii}x_i \cdot \frac{\partial \mathsf{IMM}}{\partial x_i} = 0$.
2. $\sum_{i,j\in[n]} b_{ij}x_j \cdot \frac{\partial \mathsf{IMM}}{\partial x_i} = 0$, where $b_{ij} \neq 0$ in a term only if $x_i \neq x_j$ and $x_i, x_j \in \mathbf{x}_l$ for some $l \in [d]$.
3. $\sum_{i,j\in[n]} c_{ij}x_j \cdot \frac{\partial \mathsf{IMM}}{\partial x_i} = 0$, where $c_{ij} \neq 0$ in a term only if $x_i \in \mathbf{x}_l$ and $x_j \in \mathbf{x}_k$ for $l \neq k$.

Hence every $E = (e_{ij})_{i,j\in[n]}$ in $\mathfrak{g}_{\mathsf{IMM}}$ equals $D + B + C$ where

- $D \in \mathcal{W}_1$ is a diagonal matrix,
- $B \in \mathcal{W}_2$ is a block-diagonal[39] matrix with diagonal entries zero,
- C is a matrix with nonzero entries appearing outside the above block-diagonal.

To complete the proof of the lemma we show the following.

▶ **Claim 61.** *Except those entries of* C *whose rows and columns are indexed by* $\mathbf{x}_2$ *and* $\mathbf{x}_1$ *variables respectively, or* $\mathbf{x}_{d-1}$ *and* $\mathbf{x}_d$ *variables respectively, all the other entries are zero.*

Proof. In a term $x_{pq}^{(l)} \cdot \frac{\partial \mathsf{IMM}}{\partial x_{ij}^{(k)}}$ where $l \neq k$, every monomial has two variables from $\mathbf{x}_l$ and no variable from $\mathbf{x}_k$. Hence from the equation corresponding to C we get separate equations for every pair (l, k) due to monomial disjointness:

$$\sum_{p\in[w_{l-1}],q\in[w_l]} \sum_{i\in[w_{k-1}],j\in[w_k]} c_{pq,ij}x_{pq}^{(l)} \cdot \frac{\partial \mathsf{IMM}}{\partial x_{ij}^{(k)}} = 0, \quad \text{where } l \neq k.$$

Collecting coefficients corresponding to $\frac{\partial \mathsf{IMM}}{\partial x_{ij}^{(k)}}$ in the above equation we get

$$\sum_{i\in[w_{k-1}],j\in[w_k]} \ell_{ij}^{(k)} \cdot \frac{\partial \mathsf{IMM}}{\partial x_{ij}^{(k)}} = 0, \quad \text{where } \ell_{ij}^{(k)} \text{ is a linear form in the variables from } \mathbf{x}_l. \tag{10}$$

Figure 6 depicts a term $\ell_{ij}^{(k)} \cdot \frac{\partial \mathsf{IMM}}{\partial x_{ij}^{(k)}}$ using an ABP that computes it. So the LHS of the above equation can be computed by an ABP $\mathtt{B}$ that has edge labels identical to that of the ABP for IMM, except for the edges in layer k. The (i, j)-th edge of layer k in $\mathtt{B}$ is labelled by $\ell_{ij}^{(k)}$.

[39] An entry is in the block-diagonal if and only if the variables labelling the row and column of the entry are in the same $\mathbf{x}_l$ for some $l \in [d]$.

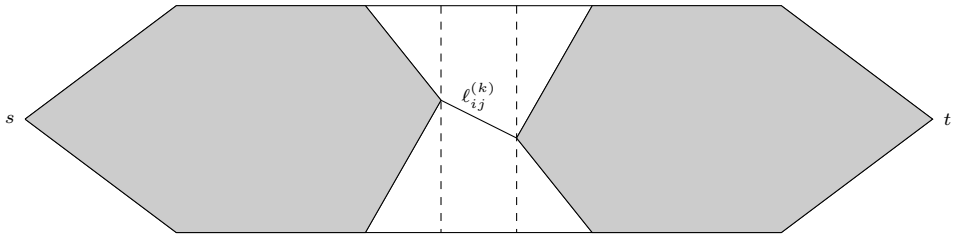


Figure 6 An ABP computing the term $\ell_{ij}^{(k)} \cdot \frac{\partial \mathsf{IMM}}{\partial x_{ij}^{(k)}}$.

Suppose $\ell_{ij}^{(k)} \neq 0$ and the coefficient of the variable $x_{pq}^{(l)}$ in $\ell_{ij}^{(k)}$ is nonzero, i.e. $c_{pq,ij} \neq 0$. If (l,k) is neither $(1,2)$ nor $(d,d-1)$ then the assumption $c_{pq,ij} \neq 0$ leads to a contradiction as follows.

Consider an s to t path P in $\mathtt{B}$ that goes through the (i,j)-th edge of layer k (which is labelled by $\ell_{ij}^{(k)}$) but excludes the (p,q)-th edge of layer l (which is labelled by $x_{pq}^{(l)}$), the (p,i)-th edge of layer $k-1$ if $l = k-1$ and the (j,q)-th edge of layer $k+1$ if $l = k+1$ (we can notice this is always possible since (l,k) is neither $(1,2)$ nor $(d,d-1)$). Then, if we retain the variables labelling the edges of P ouside the layer k and the variable $x_{pq}^{(l)}$, and set every other variable to zero then P becomes the unique s to t path in $\mathtt{B}$ with nonzero weight (since $c_{pq,ij} \neq 0$). But this contradicts the fact that ABP $\mathtt{B}$ is computing an identically zero polynomial (by Equation (10)). ◀

Therefore, $\mathfrak{g}_{\mathsf{IMM}} \subseteq \mathcal{W}_1 + \mathcal{W}_2 + \mathcal{W}_3$ implying $\mathfrak{g}_{\mathsf{IMM}} = \mathcal{W}_1 \oplus \mathcal{W}_2 \oplus \mathcal{W}_3$. ◀

▶ **Lemma 32** (restated). *The space $\mathcal{W}_3 = \mathcal{W}_3^{(a)} \oplus \mathcal{W}_3^{(b)}$ where $\mathcal{W}_3^{(a)} = \mathcal{A}_1 \oplus \mathcal{A}_2 \oplus \cdots \oplus \mathcal{A}_{w_2}$ and $\mathcal{W}_3^{(b)} = \mathcal{A'}_1 \oplus \mathcal{A'}_2 \oplus \cdots \oplus \mathcal{A'}_{w_{d-2}}$ such that for every $i \in [w_2]$ $\mathcal{A}_i$ is isomorphic to the space of $w_1 \times w_1$ anti-symmetric matrices over $\mathbb{F}$, and for every $j \in [w_{d-2}]$ $\mathcal{A'}_j$ is isomorphic to the space of $w_{d-1} \times w_{d-1}$ anti-symmetric matrices over $\mathbb{F}$. Hence* $\dim(\mathcal{W}_3) = \frac{1}{2}\left[w_1 w_2 (w_1 - 1) + w_{d-1} w_{d-2} (w_{d-1} - 1)\right]$.

Proof. Recall, $\mathcal{W}_3$ is the space of all matrices $C = (c_{ij})_{i,j \in [n]}$ such that

$$\sum_{i,j \in [n]} c_{ij} x_j \cdot \frac{\partial \mathsf{IMM}}{\partial x_i} = 0, \tag{11}$$

where in every nonzero summand either $x_i \in \mathbf{x}_2, x_j \in \mathbf{x}_1$ or $x_i \in \mathbf{x}_{d-1}, x_j \in \mathbf{x}_d$. In Equation (11) every monomial in a term $x_p^{(1)} \cdot \frac{\partial \mathsf{IMM}}{\partial x_{qr}^{(2)}}$ has two variables from $\mathbf{x}_1$. Similarly, every monomial in a term $x_p^{(d)} \cdot \frac{\partial \mathsf{IMM}}{\partial x_{qr}^{(d-1)}}$ has two variables from $\mathbf{x}_d$ respectively. Owing to monomial disjointness, Equation (11) gives two equations

$$\sum_{r \in [w_2]} \sum_{p,q \in [w_1]} c_{pqr}^{(1)} x_p^{(1)} \cdot \frac{\partial \mathsf{IMM}}{\partial x_{qr}^{(2)}} = 0, \quad \text{and} \tag{12}$$

$$\sum_{q \in [w_{d-2}]} \sum_{p,r \in [w_{d-1}]} c_{pqr}^{(d)} x_p^{(d)} \cdot \frac{\partial \mathsf{IMM}}{\partial x_{qr}^{(d-1)}} = 0. \tag{13}$$

Thus $\mathcal{W}_3 = \mathcal{W}_3^{(a)} \oplus \mathcal{W}_3^{(b)}$ where $\mathcal{W}_3^{(a)}$ consists of matrices satisfying Equation (12) and $\mathcal{W}_3^{(b)}$ consists of matrices satisfying Equation (13). We argue the following about $\mathcal{W}_3^{(a)}$.

Figure 7 An ABP computing the term $x_p^{(1)} \cdot \frac{\partial \mathsf{IMM}}{\partial x_{qr}^{(2)}}$.

Figure 8 An ABP computing the term $x_q^{(1)} \cdot \frac{\partial \mathsf{IMM}}{\partial x_{pr}^{(2)}}$.

▶ **Claim 62.** *$\mathcal{W}_3^{(a)} = \mathcal{A}_1 \oplus \mathcal{A}_2 \oplus \cdots \oplus \mathcal{A}_{w_2}$ where every $\mathcal{A}_i$ is isomorphic to the space of $w_1 \times w_1$ anti-symmetric matrices over $\mathbb{F}$.*

Proof. Figure 7 depicts an ABP computing the term $x_p^{(1)} \cdot \frac{\partial \mathsf{IMM}}{\partial x_{qr}^{(2)}}$. Every monomial in $c_{pqr}^{(1)} x_p^{(1)} \cdot \frac{\partial \mathsf{IMM}}{\partial x_{qr}^{(2)}}$ is divisible by $x_p^{(1)} x_q^{(1)}$.

The only other term in Equation (12) that contains monomials divisible by $x_p^{(1)} x_q^{(1)}$ is $c_{qpr}^{(1)} x_q^{(1)} \cdot \frac{\partial \mathsf{IMM}}{\partial x_{pr}^{(2)}}$. Figure 8 depicts an ABP computing $x_q^{(1)} \cdot \frac{\partial \mathsf{IMM}}{\partial x_{pr}^{(2)}}$.

Since the terms in Figures 7 and 8 have no monomials in common with any other term in Equation (12) it must be that $c_{pqr}^{(1)} = -c_{qpr}^{(1)}$. Moreover, if $p = q$ then $c_{pqr}^{(1)} = 0$. Thus Equation (12) gives an equation for every $r \in [w_2]$

$$\sum_{p,q \in [w_1], p \neq q} c_{pqr}^{(1)} x_p^{(1)} \cdot \frac{\partial \mathsf{IMM}}{\partial x_{qr}^{(2)}} = 0, \tag{14}$$

such that the matrix $C_r = (c_{pqr}^{(1)})_{p,q \in [w_1]} \in \mathbb{F}^{w_1 \times w_1}$ is anti-symmetric. Further any anti-symmetric matrix can be used to get an equation like Equation (14). Thus, as shown in Figure 9, every matrix $C^{(a)} \in \mathcal{W}_3^{(a)}$ is such that for every $r \in [w_2]$, the $w_1 \times w_1$ submatrix (say $C_r^{(a)}$) defined by the rows labelled by the $x_{qr}^{(2)}$ variables and the columns labelled by the $x_p^{(1)}$ variables for $p, q \in [w_1]$ is anti-symmetric.

Also, any matrix satisfying the above properties belongs to $\mathcal{W}_3^{(a)}$. Naturally, if we define $\mathcal{A}_r$ to be the space of $n \times n$ matrices such that the $w_1 \times w_1$ submatrix defined by the rows labelled by the $x_{qr}^{(2)}$ variables and the columns labelled by the $x_p^{(1)}$ variables for $p, q \in [w_1]$ is anti-symmetric and all other entries are zero then $\mathcal{W}_3^{(a)} = \mathcal{A}_1 \oplus \mathcal{A}_2 \oplus \cdots \oplus \mathcal{A}_{w_2}$. ◀

Similarly, it can be shown that $\mathcal{W}_3^{(b)} = \mathcal{A'}_1 \oplus \mathcal{A'}_2 \oplus \cdots \oplus \mathcal{A'}_{w_{d-2}}$ where every $\mathcal{A'}_i$ is isomorphic to the space of $w_{d-1} \times w_{d-1}$ anti-symmetric matrices. This completes the proof of Lemma 32. ◀

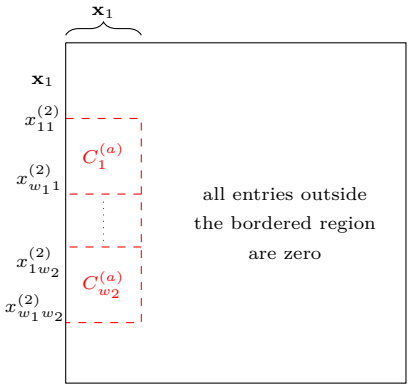


Figure 9 A matrix $C^{(a)}$ in $\mathcal{W}_3^{(a)}$.

▶ **Lemma 33** (restated). *The space $\mathcal{W}_2 = \mathcal{B}_1 \oplus \mathcal{B}_2 \oplus \cdots \oplus \mathcal{B}_{d-1}$ such that for every $k \in [d-1]$, $\mathcal{B}_k$ is isomorphic to the $\mathbb{F}$-linear space spanned by $t_k \times t_k$ matrices of the form*

$$\begin{bmatrix} -Z^T \otimes I_{w_{k-1}} & 0 \\ 0 & I_{w_{k+1}} \otimes Z \end{bmatrix}_{t_k \times t_k} \quad \text{where } Z \in \mathcal{Z}_{w_k} \text{ and } t_k = w_k(w_{k-1} + w_{k+1}).$$

Hence, $\dim(\mathcal{W}_2) = \sum_{k=1}^{d-1}(w_k^2 - w_k)$.

Proof. Recall $w_0 = w_d = 1$ and $\mathcal{Z}_{w_k}$ denotes the space of $w_k \times w_k$ matrix with diagonal entries 0, and $\mathcal{W}_2$ is the space of all matrices $B = (b_{ij})_{i,j\in[n]}$ such that

$$\sum_{i,j\in[n]} b_{ij} x_j \cdot \frac{\partial \mathsf{IMM}}{\partial x_i} = 0, \tag{15}$$

where in every term $b_{ij} \neq 0$ only if $x_i \neq x_j$ and $x_i, x_j \in \mathbf{x}_l$ for some $l \in [d]$. The following observation is easy to verify.

▶ **Observation 63.** *Suppose $l \in [2, d-1]$. A term $x^{(l)}_{i_1 j_1} \cdot \frac{\partial \mathsf{IMM}}{\partial x^{(l)}_{i_2 j_2}}$ where $i_1 \neq i_2$ and $j_1 \neq j_2$ does not share a monomial with any other term in Equation (15).*

Hence for $l \in [2, d-1]$, terms of the kind $x^{(l)}_{i_1 j_1} \cdot \frac{\partial \mathsf{IMM}}{\partial x^{(l)}_{i_2 j_2}}$ where $i_1 \neq i_2$ and $j_1 \neq j_2$ are absent in Equation (15). A monomial appearing in a nonzero term of Equation (15) is of the form $x^{(1)}_{i_1} \cdot x^{(2)}_{i_1 i_2} \cdots x^{(k)}_{i_{k-1} i_k} \cdot x^{(k+1)}_{i'_k i_{k+1}} \cdots x^{(d-1)}_{i_{d-1} i_d} \cdot x^{(d)}_{i_d}$ where $i_k \neq i'_k$, for some $k \in [d-1]$. We say such a monomial is broken at the k-th interface. Observe the following.

▶ **Observation 64.** *The terms $x^{(k)}_{pr} \cdot \frac{\partial \mathsf{IMM}}{\partial x^{(k)}_{pq}}$ where $p \in [w_{k-1}]$, $q, r \in [w_k]$, $q \neq r$, and $x^{(k+1)}_{mj} \cdot \frac{\partial \mathsf{IMM}}{\partial x^{(k+1)}_{ij}}$ where $i, m \in [w_k]$, $j \in [w_{k+1}]$, $i \neq m$ are the only two whose monomials are broken at the k-th interface.*

Thus from Equation (15) we get $(d-1)$ equations one for each interface by considering cancellations of monomials broken at that interface. For $k \in [2, d-2]$, let $\mathcal{B}_k$ be the space of all $n \times n$ matrices B_k such that

1. the entry corresponding to the row labelled by $x^{(k)}_{pq}$ and the column labelled by $x^{(k)}_{pr}$ is $b^{(k)}_{pq,pr} \in \mathbb{F}$ for $p \in [w_{k-1}], q, r \in [w_k]$ and $q \neq r$,
2. the entry corresponding to the row labelled by $x^{(k+1)}_{ij}$ and the column labelled by $x^{(k+1)}_{mj}$ is $b^{(k+1)}_{ij,mj} \in \mathbb{F}$ for $i, m \in [w_k], j \in [w_{k+1}]$ and $i \neq m$,

3. all other entries of B_k are zero, and
4.

$$\sum_{p\in[w_{k-1}],\ q,r\in[w_k],\ q\neq r} b^{(k)}_{pq,pr}x^{(k)}_{pr}\cdot\frac{\partial \mathsf{IMM}}{\partial x^{(k)}_{pq}} \ + \sum_{i,m\in[w_k],\ j\in[w_{k+1}],\ i\neq m} b^{(k+1)}_{ij,mj}x^{(k+1)}_{mj}\cdot\frac{\partial \mathsf{IMM}}{\partial x^{(k+1)}_{ij}} \ = \ 0. \tag{16}$$

We can define spaces $\mathcal{B}_1$ and $\mathcal{B}_{d-1}$ similarly considering monomials broken at the first and the last interface respectively. As Equation (15) can be split into $(d-1)$ equations, one for every interface, $\mathcal{W}_2 = \mathcal{B}_1 + \mathcal{B}_2 + \cdots + \mathcal{B}_{d-1}$. Since the spaces $\mathcal{B}_1, \ldots, \mathcal{B}_{d-1}$ control different entries of $n \times n$ matrices, $\mathcal{W}_2 = \mathcal{B}_1 \oplus \mathcal{B}_2 \oplus \cdots \oplus \mathcal{B}_{d-1}$.

▶ **Claim 65.** *For $k \in [2, d-2]$, $\mathcal{B}_k$ is isomorphic to the $\mathbb{F}$-linear space spanned by $t_k \times t_k$ matrices of the form*

$$\begin{bmatrix} -Z^T \otimes I_{w_{k-1}} & 0 \\ 0 & I_{w_{k+1}} \otimes Z \end{bmatrix}_{t_k \times t_k} \quad \textit{where } Z \in \mathcal{Z}_{w_k} \textit{ and } t_k = w_k(w_{k-1}+w_{k+1}).$$

Proof. Collecting same derivative terms in Equation (16) we get

$$\sum_{p\in[w_{k-1}],q\in[w_k]} \ell^{(k)}_{pq}\cdot\frac{\partial \mathsf{IMM}}{\partial x^{(k)}_{pq}} \ + \sum_{i\in[w_k],j\in[w_{k+1}]} \ell^{(k+1)}_{ij}\cdot\frac{\partial \mathsf{IMM}}{\partial x^{(k+1)}_{ij}} = 0, \tag{17}$$

where $\ell^{(k)}_{pq}$ is a linear form containing variables $x^{(k)}_{pr}$ such that $r \neq q$, and $\ell^{(k+1)}_{ij}$ is a linear form containing variables $x^{(k+1)}_{mj}$ such that $m \neq i$. Here is a succinct way to write Equation (17):

$$Q_1\cdot Q_2\cdots Q_k^{'}\cdot Q_{k+1}\cdot Q_{k+2}\cdots Q_{d-1}\cdot Q_d \ + \ Q_1\cdot Q_2\cdots Q_k\cdot Q_{k+1}^{'}\cdot Q_{k+2}\cdots Q_{d-1}\cdot Q_d = 0, \tag{18}$$

where $Q_1, \ldots, Q_d$ are matrices as in Section 2.3, $Q_k^{'} = (\ell^{(k)}_{pq})_{p\in[w_{k-1}],q\in[w_k]}$ and $Q_{k+1}^{'} = (\ell^{(k+1)}_{ij})_{i\in[w_k],j\in[w_{k+1}]}$. This implies

$$Q_k^{'}\cdot Q_{k+1} \ + \ Q_k\cdot Q_{k+1}^{'} = 0,$$

as $Q_1, \ldots, Q_d$ have distinct sets of variables, and the variables appearing in $Q_k^{'}$ and $Q_{k+1}^{'}$ are the same as in Q_k and Q_{k+1} respectively. The variable disjointness of Q_k and Q_{k+1} can be exploited to infer $Q_{k+1}^{'} = Z\cdot Q_{k+1}$ and $Q_k^{'} = -Q_k\cdot Z$ where Z is in $\mathbb{F}^{w_k\times w_k}$ (even if Q_k, Q_{k+1} may not be square matrices). As the linear form $\ell^{(k)}_{pq}$ is devoid of the variable $x^{(k)}_{pq}$, it must be that $Z \in \mathcal{Z}_{w_k}$. Moreover, any $Z \in \mathcal{Z}_{w_k}$ can be used along with the relations $Q_{k+1}^{'} = Z\cdot Q_{k+1}$ and $Q_k^{'} = -Q_k\cdot Z$ to satisfy Equation (18) and hence also Equations (16) and (17).

Let $Z = (z_{im})_{i,m\in[w_k]}$. Since $Q_{k+1}^{'} = Z\cdot Q_{k+1}$, the coefficient of $x^{(k+1)}_{mj}$ in $\ell^{(k+1)}_{ij}$ is z_{im} for every $j \in [w_{k+1}]$. Hence in Equation (16), $b^{(k+1)}_{ij,mj} = z_{im}$ for every $j \in [w_{k+1}]$. Similarly, since $Q_k^{'} = -Q_k\cdot Z$ the coefficient of $x^{(k)}_{pr}$ in $\ell^{(k)}_{pq}$ is $-z_{rq}$ for every $p \in [w_{k-1}]$. Hence in Equation (16) $b^{(k)}_{pq,pr} = -z_{rq}$ for every $p \in [w_{k-1}]$. Thus the submatrix of B_k defined by the rows and columns labelled by the variables in $\mathbf{x}_k$ and $\mathbf{x}_{k+1}$ looks like

$$\begin{bmatrix} -Z^T \otimes I_{w_{k-1}} & 0 \\ 0 & I_{w_{k+1}} \otimes Z \end{bmatrix}_{t_k \times t_k}$$

where $t_k = w_k(w_{k-1}+w_{k+1})$ and all other entries in B_k are zero. Hence $\mathcal{B}_k$ is isomorphic to the space generated by $t_k \times t_k$ matrices of the above kind. This proves the claim. ◀

We can similarly show that $\mathcal{B}_1$ is isomorphic to the space generated by square matrices of the form

$$\begin{bmatrix} -Z^T & 0 \\ 0 & I_{w_2} \otimes Z \end{bmatrix}_{t_1 \times t_1} \quad \text{where } Z \in \mathcal{Z}_{w_1} \text{ and } t_1 = w_1 + w_1 w_2,$$

and $\mathcal{B}_{d-1}$ is isomorphic to the space generated by square matrices of the form

$$\begin{bmatrix} -Z^T \otimes I_{w_{d-2}} & 0 \\ 0 & Z \end{bmatrix}_{t_{d-1} \times t_{d-1}} \quad \text{where } Z \in \mathcal{Z}_{w_{d-1}} \text{ and } t_{d-1} = w_{d-1} w_{d-2} + w_{d-1}.$$

This completes the proof of Lemma 33. ◀

▶ **Lemma 34** (restated). *The space $\mathcal{W}_1$ contains the space $\mathcal{D}_1 \oplus \mathcal{D}_2 \oplus \cdots \oplus \mathcal{D}_{d-1}$ such that for every $k \in [d-1]$, $\mathcal{D}_k$ is isomorphic to the $\mathbb{F}$-linear space spanned by $t_k \times t_k$ matrices of the form*

$$\begin{bmatrix} -Y \otimes I_{w_{k-1}} & 0 \\ 0 & I_{w_{k+1}} \otimes Y \end{bmatrix}_{t_k \times t_k} \quad \textit{where } Y \in \mathcal{Y}_{w_k} \textit{ and } t_k = w_k(w_{k-1} + w_{k+1}).$$

Hence, $\dim(\mathcal{W}_1) \geq \sum_{k=1}^{d-1} w_k$.

Proof. The proof is similar to the proof of Lemma 33. Recall $w_0 = w_d = 1$ and $\mathcal{Y}_{w_k}$ denotes the space of $w_k \times w_k$ diagonal matrices. Every $D \in \mathcal{W}_1$ satisfies an equation of the following form

$$\sum_{i \in [w_1]} d_i^{(1)} x_i^{(1)} \cdot \frac{\partial \mathsf{IMM}}{\partial x_i^{(1)}} + \sum_{k=2}^{d-1} \sum_{i \in [w_{k-1}], j \in [w_k]} d_{ij}^{(k)} x_{ij}^{(k)} \cdot \frac{\partial \mathsf{IMM}}{\partial x_{ij}^{(k)}} + \sum_{i \in [w_{d-1}]} d_i^{(d)} x_i^{(d)} \cdot \frac{\partial \mathsf{IMM}}{\partial x_i^{(d)}} = 0.$$

A succinct way to write the above equation is

$$\sum_{k=1}^{d} Q_1 Q_2 \cdots Q_{k-1} Q_k' Q_{k+1} \cdots Q_d = 0, \tag{19}$$

where $Q_1' = (d_i^{(1)} x_i^{(1)})_{i \in [w_1]}$ is a row vector, $Q_d' = (d_i^{(d)} x_i^{(d)})_{i \in [w_{d-1}]}^T$ is a column vector, $Q_k' = (d_{ij}^{(k)} x_{ij}^{(k)})_{i \in [w_{k-1}], j \in [w_k]}$, and $Q_1, \ldots, Q_d$ are matrices as in Section 2.3. For every $k \in [d-1]$, let us focus on those $D_k \in \mathcal{W}_1$ for which the matrices $Q_1', \ldots, Q_{k-1}', Q_{k+2}', \ldots, Q_d'$ are zero in Equation (19). Such a D_k satisfies the following equation,

$$Q_1 \cdot Q_2 \cdot \cdots \cdot Q_k' \cdot Q_{k+1} \cdots Q_d + Q_1 \cdot Q_2 \cdots Q_k \cdot Q_{k+1}' \cdots Q_d = 0. \tag{20}$$

Using a similar argument as in the proof of Lemma 33 we get $Q_{k+1}' = Y \cdot Q_{k+1}$ and $Q_k' = -Q_k \cdot Y$ where $Y \in \mathcal{Y}_{w_k}$. Further, any $Y \in \mathcal{Y}_{w_k}$ can be used along with the relations $Q_{k+1}' = Y \cdot Q_{k+1}$ and $Q_k' = -Q_k \cdot Y$ to satisfy Equation (20). The set of $D_k \in \mathcal{W}_1$ satisfying Equation (20) forms an $\mathbb{F}$-linear space; call it $\mathcal{D}_k$. Every $D_k \in \mathcal{D}_k$ is such that the submatrix defined by the rows and the columns labelled by the variables in $\mathbf{x}_k$ and $\mathbf{x}_{k+1}$ looks like

$$\begin{bmatrix} -Y \otimes I_{w_{k-1}} & 0 \\ 0 & I_{w_{k+1}} \otimes Y \end{bmatrix}_{t_k \times t_k} \quad \text{where } Y \in \mathcal{Y}_{w_k} \text{ and } t_k = w_k(w_{k-1} + w_{k+1}),$$

and all other entries in D_k are zero. Moreover, any $n \times n$ matrix with this structure is in $\mathcal{D}_k$. Thus $\mathcal{D}_k$ is isomorphic to the space of all $t_k \times t_k$ matrices of the form shown above. It can also be easily verified that every matrix in $\mathcal{D}_1 + \ldots + \mathcal{D}_{d-1}$ can be expressed *uniquely* as a sum of matrices in these spaces. Hence $\mathcal{W}_1 \supseteq \mathcal{D}_1 \oplus \mathcal{D}_2 \oplus \cdots \oplus \mathcal{D}_{d-1}$ completing the proof of Lemma 34. ◀

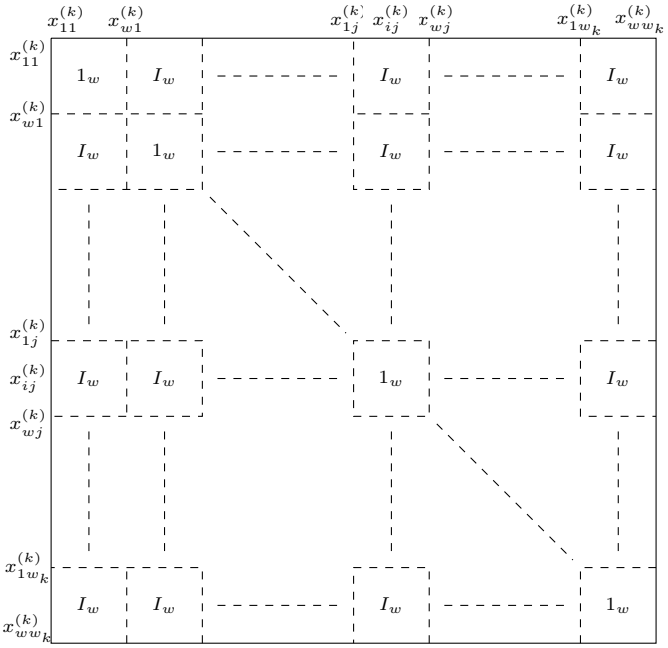


Figure 10 Submatrix of L restricted to rows/columns indexed by $\mathbf{x}_k$.

▶ **Claim 40** (restated). *No invariant subspace of $\mathfrak{g}_{\mathsf{IMM}}$ is properly contained in $\mathcal{U}_k$ for $k \in [2, d-1]$.*

Proof. Let $\mathcal{U} \subseteq \mathcal{U}_k$ be an invariant subspace of $\mathfrak{g}_{\mathsf{IMM}}$. From Claim 38 it follows that $\mathcal{U}$ is a coordinate subspace. For $t \in \mathbb{N}$, let $\tilde{1}_t \stackrel{\text{def}}{=} 1_t - I_t$, where 1_t is the $t \times t$ all one matrix. From Lemma 33, there are matrices B_{k-1} and B_k in $\mathfrak{g}_{\mathsf{IMM}}$ such that the submatrix of B_{k-1} restricted to the rows and the columns labelled by the variables in $\mathbf{x}_{k-1} \uplus \mathbf{x}_k$ looks like

$$\begin{bmatrix} -\tilde{1}_{w_{k-1}} \otimes I_{w_{k-2}} & 0 \\ 0 & I_{w_k} \otimes \tilde{1}_{w_{k-1}} \end{bmatrix}, \text{ and}$$

the submatrix in B_k restricted to the rows and the columns labelled by the variables in $\mathbf{x}_k \uplus \mathbf{x}_{k+1}$ looks like

$$\begin{bmatrix} \tilde{1}_{w_k} \otimes I_{w_{k-1}} & 0 \\ 0 & I_{w_{k+1}} \otimes -\tilde{1}_{w_k} \end{bmatrix}.$$

From Lemma 34, there is a diagonal matrix D_{k-1} in $\mathfrak{g}_{\mathsf{IMM}}$ such that the submatrix restricted to the rows and the columns labelled by the variables in $\mathbf{x}_{k-1} \uplus \mathbf{x}_k$ looks like

$$\begin{bmatrix} -I_{w_{k-1}} \otimes I_{w_{k-2}} & 0 \\ 0 & I_{w_k} \otimes I_{w_{k-1}} \end{bmatrix}.$$

Let $L = B_{k-1} + B_k + D_{k-1}$. The submatrix of L restricted to the rows and the columns labelled by the variables in $\mathbf{x}_k$ looks as shown in Figure 10.

For notational simplicity we write w_{k-1} as w in Figure 10. If e_x is a unit vector in $\mathcal{U}$, where $x = x_{ij}^{(k)}$ is a variable in $\mathbf{x}_k$ then the matrix L maps e_x to Le_x which is the column of L labelled by the variable x. This column vector has all entries zero except for the rows labelled by the variables in $\mathbf{x}_k$. Restricting to these rows and looking at Figure 10, we infer that the

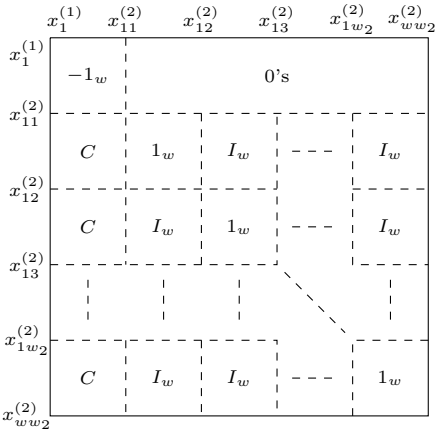


Figure 11 Submatrix of M matrix restricted to rows/columns indexed by $\mathbf{x}_1 \uplus \mathbf{x}_2$.

rows of Le_x labelled by the variables $x_{1j}^{(k)}, x_{2j}^{(k)}, \ldots, x_{w_{k-1}j}^{(k)}$ are 1 (in particular, these entries are nonzero). We use this knowledge and that $Le_x \in \mathcal{U}$ to make the following observation, the proof of which is immediate from Claim 38.

▶ **Observation 66.** *If $e_x \in \mathcal{U}$, where $x = x_{ij}^{(k)}$ then $e_{x'} \in \mathcal{U}$ for every $x' \in \{x_{1j}^{(k)}, x_{2j}^{(k)}, \ldots, x_{w_{k-1}j}^{(k)}\}$.*

Moreover, it follows from the presence of I_w matrices in Figure 10 that for every $j' \in [w_k]$ there is the variable $y = x_{ij'}^{(k)}$ such that the row labelled by y in Le_x is 1, implying[40] $e_y \in \mathcal{U}$. Hence from Observation 66, $e_{y'} \in \mathcal{U}$ for every $y' \in \{x_{1j'}^{(k)}, \ldots, x_{w_{k-1}j'}^{(k)}\}$. Since this is true for every $j' \in [w_k]$, $e_y \in \mathcal{U}$ for every variable $y \in \mathbf{x}_k$ implying $\mathcal{U} = \mathcal{U}_k$. ◀

▶ **Claim 41** (restated). *The invariant subspaces $\mathcal{U}_{1,2}$ and $\mathcal{U}_{d-1,d}$ are irreducible, and the only invariant subspace properly contained in $\mathcal{U}_{1,2}$ (respectively $\mathcal{U}_{d-1,d}$) is $\mathcal{U}_2$ (respectively $\mathcal{U}_{d-1}$).*

Proof. We prove the claim for $\mathcal{U}_{1,2}$, the proof for $\mathcal{U}_{d-1,d}$ is similar. Suppose $\mathcal{U}_{1,2} = \mathcal{V} \oplus \mathcal{W}$ where $\mathcal{V}$,$\mathcal{W}$ are invariant subspaces of $\mathfrak{g}_{\mathsf{IMM}}$ (and so also coordinate subspaces). A unit vector e_x, where $x \in \mathbf{x}_1$ is either in $\mathcal{V}$ or $\mathcal{W}$. Suppose $e_x \in \mathcal{V}$; we will show that $\mathcal{V} = \mathcal{U}_{1,2}$. Without loss of generality, let $x = x_1^{(1)}$. Arguing as in the proof of the previous claim, we infer that there is a matrix $M \in \mathfrak{g}_{\mathsf{IMM}}$ such that the submatrix of M restricted to the rows and the columns labelled by the variables in $\mathbf{x}_1$ and $\mathbf{x}_2$ looks as shown in Figure 11, in which $w = w_1$ and C is a $w_1 \times w_1$ anti-symmetric matrix with all non-diagonal entries nonzero. All the other entries of M are zero.

The vector Me_x is the first column of M and it is zero everywhere except for the rows labelled by the variables in $\mathbf{x}_1 \uplus \mathbf{x}_2$. Among these rows, unless $y \in \{x_{11}^{(2)}, x_{12}^{(2)}, \ldots, x_{1w_2}^{(2)}\}$ the row of Me_x labelled by y is nonzero. Thus (from Claim 38), $e_y \in \mathcal{V}$ for $y \in \mathbf{x}_1$ and $y = x_{ij}^{(2)}$ where $i \in [2, w_1]$ and $j \in [w_2]$. Let $y = x_{ij}^{(2)}$ for some $i \in [2, w_1]$ and $j \in [w_2]$. From Figure 11, the row of Me_y labelled by $x_{1j}^{(2)}$ is nonzero and so, for $y' = x_{1j}^{(2)}$, $e_{y'}$ is also in $\mathcal{V}$. Hence, $\mathcal{V} = \mathcal{U}_{1,2}$ and $\mathcal{U}_{1,2}$ is irreducible. To argue that the only invariant subspace properly contained in $\mathcal{U}_{1,2}$ is $\mathcal{U}_2$, let $\mathcal{V} \subset \mathcal{U}_{1,2}$ be an invariant subspace of $\mathfrak{g}_{\mathsf{IMM}}$. From the above argument it follows that $e_x \notin \mathcal{V}$ for every $x \in \mathbf{x}_1$ (otherwise $\mathcal{V} = \mathcal{U}_{1,2}$). This implies $\mathcal{V} \subseteq \mathcal{U}_2$ and from Claim 40 we have $\mathcal{V} = \mathcal{U}_2$. ◀

[40] Follows again from Claim 38.

7.3 Proof of claims in Section 4

▶ **Claim 43** (restated). *For all $i \in [s]$, let $\mathcal{N}_i$ and $\mathcal{N}_i^{'}$ be the null spaces of $g_i(R)$ and $g_i(R^{'})$. Then*
1. $\mathbb{F}^n = \mathcal{N}_1 \oplus \mathcal{N}_2 \oplus \cdots \oplus \mathcal{N}_s = \mathcal{N}_1^{'} \oplus \mathcal{N}_2^{'} \oplus \cdots \oplus \mathcal{N}_s^{'}$.
2. *For all $i \in [s]$, $\dim(\mathcal{N}_i) = \dim(\mathcal{N}_i^{'}) = \deg_x(g_i)$.*

Proof. Since $\mathcal{N}_i^{'} = A^{-1}\mathcal{N}_i$ and $A^{-1} \in \mathsf{GL}(n)$, it is sufficient to show $\mathbb{F}^n = \mathcal{N}_1 \oplus \mathcal{N}_2 \oplus \cdots \oplus \mathcal{N}_s$ and $\dim(\mathcal{N}_i) = \deg_x(g_i)$. Further, observe that each subspace $\mathcal{N}_i$ is non-trivial – if $\mathcal{N}_1 = \{0\}$ then for all $\mathbf{v} \in \mathbb{F}^n$, $h(R) \cdot \mathbf{v} = g_1(R)g_2(R)\cdots g_s(R) \cdot \mathbf{v} = 0$ implying $g_2(R)\cdots g_s(R) \cdot \mathbf{v} = 0$. As the characteristic polynomial and the minimal polynomial have the same irreducible factors this gives a contradiction.

To show the sum of $\mathcal{N}_i$'s is a direct sum it is sufficient to show the following: if $\sum_{l=1}^{s} \mathbf{u}_l = 0$ where $\mathbf{u}_l \in \mathcal{N}_l$ then $\mathbf{u}_l = 0$ for $l \in [s]$. Define for $i \in [s]$

$$\hat{g}_i \coloneqq \prod_{j=1, j\neq i}^{s} g_j(x) = \frac{h(x)}{g_i(x)}. \tag{21}$$

Since $\hat{g}_i(R) \cdot \mathbf{u}_j = 0$ for $j \neq i$,

$$\hat{g}_i(R) \cdot \left(\sum_{l=1}^{s} \mathbf{u}_l \right) = \hat{g}_i(R) \cdot \mathbf{u}_i = 0. \tag{22}$$

As $g_i(x)$ and $\hat{g}_i(x)$ are coprime polynomials, there are $p_i(x), q_i(x) \in \mathbb{F}[x]$ such that

$$p_i(x)g_i(x) + q_i(x)\hat{g}_i(x) = 1 \quad \Rightarrow \quad p_i(R)g_i(R) + q_i(R)\hat{g}_i(R) = I_n$$

$$\Rightarrow \quad (p_i(R)g_i(R)) \cdot \mathbf{u}_i + (q_i(R)\hat{g}_i(R)) \cdot \mathbf{u}_i = \mathbf{u}_i.$$

Both $(p_i(R)g_i(R)) \cdot \mathbf{u}_i = 0$ (as $\mathbf{u}_i \in \mathcal{N}_i$) and $(q_i(R)\hat{g}_i(R)) \cdot \mathbf{u}_i = 0$ (by Equation (22)). Hence $\mathbf{u}_i = 0$ for all $i \in [s]$.

Let $\tilde{R}$ be the linear the linear map R restricted to the subspace $\mathcal{N}_i$ (this is well defined as $\mathcal{N}_i$ is an invariant subspace of R). Then, $g_i(\tilde{R}) = 0$. Since g_i is irreducible, from Cayley-Hamilton theorem it follows that g_i divides the characteristic polynomial of $\tilde{R}$ implying $\deg_x(g_i) \leq \dim(\mathcal{N}_i)$. As a consequence, we have

$$n = \sum_{i=1}^{s} \deg_x g_i \leq \sum_{i=1}^{s} \dim \mathcal{N}_i \leq \dim \mathbb{F}^n = n. \tag{23}$$

Each inequality is an equality, which proves the claim. ◀

▶ **Claim 44** (restated). *Suppose $g_i(x)$ is an irreducible factor of the characteristic polynomial $h_k(x)$ of R_k (depicted in Figure 4) for some $k \in [d]$. Then the following holds:*
1. *If $k \in [2, d-1]$ then $\mathcal{N}_i \subseteq \mathcal{U}_k$ (equivalently $\mathcal{N}_i^{'} \subseteq A^{-1}\mathcal{U}_k$).*
2. *If $k = 1$ then $\mathcal{N}_i \subseteq \mathcal{U}_{1,2}$ (equivalently $\mathcal{N}_i^{'} \subseteq A^{-1}\mathcal{U}_{1,2}$), and if $k = d$ then $\mathcal{N}_i \subseteq \mathcal{U}_{d-1,d}$ (equivalently $\mathcal{N}_i^{'} \subseteq A^{-1}\mathcal{U}_{d-1,d}$).*

Proof. Figure 12 depicts the matrix $h_k(R)$ and as shown in it, call the submatrix restricted to the rows labelled by variables in $\mathbf{x}_2$ and columns labelled by variables in $\mathbf{x}_1 \uplus \mathbf{x}_2$, $M_{k,2}$; define $M_{k,d-1}$ similarly.

Let $\mathbf{v} \in \mathcal{N}_i$. For every $j \in [d]$, let $\mathbf{v}_j$ be the subvector of $\mathbf{v}$ restricted to the rows labelled by variables in $\mathbf{x}_j$, and $\mathbf{v}_{1,2}$ (respectively $\mathbf{v}_{d-1,d}$) be the subvector of $\mathbf{v}$ restricted to the rows

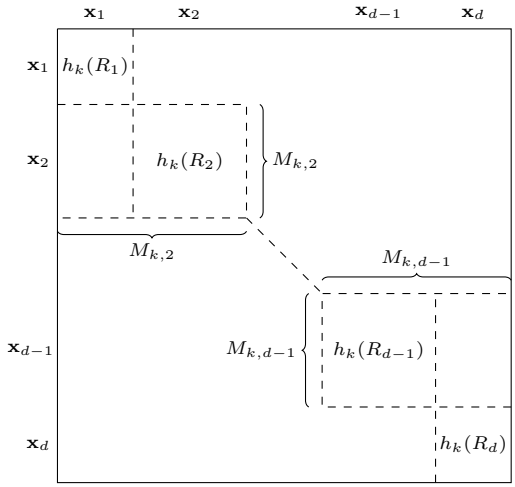


Figure 12 Matrix $h_k(R)$.

labelled by variables in $\mathbf{x}_1 \uplus \mathbf{x}_2$ (respectively $\mathbf{x}_{d-1} \uplus \mathbf{x}_d$). Since $\mathbf{v} \in \mathcal{N}_i$, $g_i(R) \cdot \mathbf{v} = 0$ implying $h_k(R) \cdot \mathbf{v} = 0$. Thus we have the following set of equations:

$$\begin{aligned} h_k(R_1) \cdot \mathbf{v}_1 &= 0 \\ M_{k,2} \cdot \mathbf{v}_{1,2} &= 0 \\ h_k(R_j) \cdot \mathbf{v}_j &= 0 \quad \text{for } j \in [3, d-2] \\ M_{k,d-1} \cdot \mathbf{v}_{d-1,d} &= 0 \\ h_k(R_d) \cdot \mathbf{v}_d &= 0. \end{aligned} \tag{24}$$

Case A: $k \in [2, d-1]$: Since $h_j(x)$ is the characteristic polynomial of R_j, $h_j(R_j) = 0$ implying $h_j(R_j) \cdot \mathbf{v}_j = 0$ for every $j \in [d]$. As $k \neq 1$, $h_k(x)$ and $h_1(x)$ are coprime and from Equation (24) $h_k(R_1) \cdot \mathbf{v}_1 = 0$. Hence, $\mathbf{v}_1 = 0$ and for a similar reason $\mathbf{v}_d = 0$ as $k \neq d$. Thus in Equation (24) we have

$$\begin{aligned} M_{k,2} \cdot \mathbf{v}_{1,2} = \; & h_k(R_2) \cdot \mathbf{v}_2 \;\; = \; 0 \\ M_{k,d-1} \cdot \mathbf{v}_{d-1,d} = \; & h_k(R_{d-1}) \cdot \mathbf{v}_{d-1} \;\; = \; 0. \end{aligned}$$

Therefore for every $j \in [d]$, $h_k(R_j) \cdot \mathbf{v}_j = 0$. If $j \neq k$ then $h_j(x)$ and $h_k(x)$ are coprime, thus from $h_j(R_j) \cdot \mathbf{v}_j = 0$ we infer $\mathbf{v}_j = 0$ and hence $\mathbf{v} \in \mathcal{U}_k$.

Case B: $k = 1$ or $k = d$: Let $k = 1$, the proof for $k = d$ is similar. Since $h_k(R_d) \cdot \mathbf{v}_d = 0$, $h_d(R_d) \cdot \mathbf{v}_d = 0$, and $h_k(x)$, $h_d(x)$ are coprime, we get $\mathbf{v}_d = 0$. Hence from Equation (24),

$$M_{k,d-1} \cdot \mathbf{v}_{d-1,d} = \; h_k(R_{d-1}) \cdot \mathbf{v}_{d-1} = 0.$$

Again for $j \in [3, d]$, $h_k(R_j) \cdot \mathbf{v}_j = 0$ and $h_j(x)$, $h_k(x)$ are coprime for every $j \neq k$. Hence $\mathbf{v}_j = 0$ for $j \in [3, d]$ implying $\mathbf{v} \in \mathcal{U}_{1,2}$. ◀

7.4 Proof of lemma and claim in Section 5

▶ **Lemma 49** (restated). *If $f = X_1 \cdot X_2 \cdots X_d$ and $\mathcal{Y}_1, \mathcal{Y}_{1,2}, \mathcal{Y}_3, \ldots, \mathcal{Y}_{d-2}, \mathcal{Y}_{d-1,d}, \mathcal{Y}_d$ is the output of Algorithm 5 then there is a permutation σ on $[3, d-2]$ such that the following hold:*

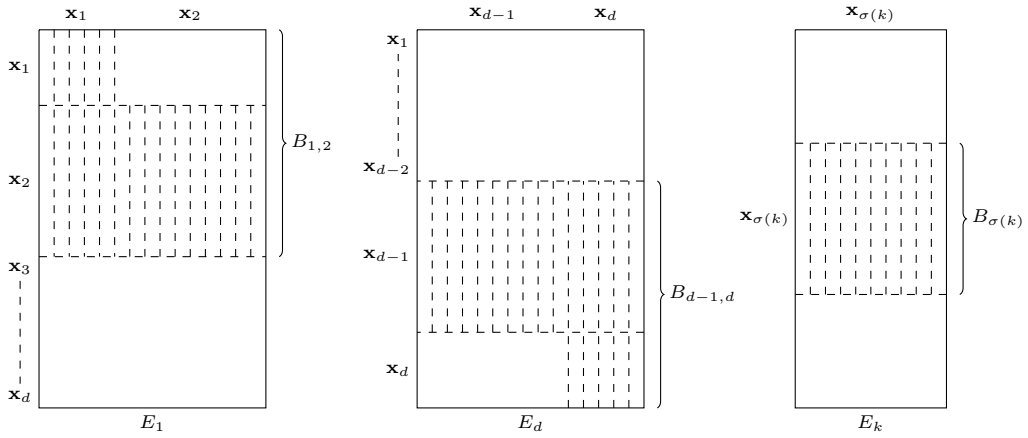


Figure 13 Matrices E_1, E_d and E_k.

1. *For every* $k \in [3, d-2]$, $\mathcal{Y}_k = \mathcal{X}_{\sigma(k)}$.
2. *Either* $\mathcal{Y}_1, \mathcal{Y}_{1,2}$ *and* $\mathcal{Y}_d, \mathcal{Y}_{d-1,d}$ *are* $\mathcal{X}_1, \mathcal{X}_{1,2}$ *and* $\mathcal{X}_d, \mathcal{X}_{d-1,d}$ *respectively, or* $\mathcal{Y}_1, \mathcal{Y}_{1,2}$ *and* $\mathcal{Y}_d$, $\mathcal{Y}_{d-1,d}$ *are* $\mathcal{X}_d, \mathcal{X}_{d-1,d}$ *and* $\mathcal{X}_1, \mathcal{X}_{1,2}$ *respectively.*

Proof. Assume $\mathcal{V}_1$ and $\mathcal{V}_d$ are the spaces $A^{-1}\mathcal{U}_{1,2}$ and $A^{-1}\mathcal{U}_{d-1,d}$ respectively. In this case we will show $\mathcal{Y}_1, \mathcal{Y}_{1,2}$ and $\mathcal{Y}_d, \mathcal{Y}_{d-1,d}$ are $\mathcal{X}_1, \mathcal{X}_{1,2}$ and $\mathcal{X}_d, \mathcal{X}_{d-1,d}$ respectively[41]. Hence, $u_1 = w_1 + w_1w_2$, $u_2 = w_1w_2$, $u_{d-1} = w_{d-2}w_{d-1}$ and $u_d = w_{d-1} + w_{d-2}w_{d-1}$. From the order of the columns in V_1 and V_d we have $V_1 = A^{-1}E_1$ and $V_d = A^{-1}E_d$, where E_1 and E_d are $n \times u_1$ and $n \times u_d$ matrices respectively and they look as shown in Figure 13.

The rows of E_1 and E_d are labelled by n variables in $\mathbf{x}_1$ to $\mathbf{x}_d$, whereas the columns of E_1 are labelled by variables in $\mathbf{x}_1$ and $\mathbf{x}_2$ and the columns of E_d are labelled by variables in $\mathbf{x}_{d-1}$ and $\mathbf{x}_d$. Moreover, the nonzero entries in these matrices are restricted to the shaded region in Figure 13.

For $k \in [3, d-2]$, $\mathcal{V}_k = A^{-1}\mathcal{U}_{\sigma(k)}$ where σ is a permutation on $[3, d-2]$. Hence, $u_k = w_{\sigma(k)-1}w_{\sigma(k)}$ and $V_k = A^{-1}E_k$ where E_k is a $n \times u_k$ matrix and looks as shown in Figure 13. Again the rows of E_k are labelled by the variables $\mathbf{x}_1$ to $\mathbf{x}_d$, whereas the columns of E_k are labelled by variables in $\mathbf{x}_{\sigma(k)}$. The nonzero entries in E_k are restricted to the shaded region in Figure 13 whose rows are labelled by variables in $\mathbf{x}_{\sigma(k)}$. Let E be the concatenation of these matrices, $E = [E_1 \mid E_3 \mid E_4 \mid \ldots \mid E_{d-2} \mid E_d]$. The rows of E are labelled by $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_d$ as usual , but now the columns are labelled by $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_{\sigma(3)}$, $\ldots, \mathbf{x}_{\sigma(d-2)}, \mathbf{x}_{d-1}, \mathbf{x}_d$ in order as shown in Figure 14. Then $V = A^{-1}E$ implying $V^{-1} = E^{-1}A$. Owing to the structure of E, E^{-1} looks as shown in Figure 14.

The rows of E^{-1} are labelled by $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_{\sigma(3)}, \ldots, \mathbf{x}_{\sigma(d-2)}, \mathbf{x}_{d-1}, \mathbf{x}_d$ in order, whereas the columns are labelled by the usual ordering $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_d$. The submatrix of E^{-1} restricted to the rows and columns labelled by the variables in $\mathbf{x}_1$ and $\mathbf{x}_2$ is $B_{1,2}^{-1}$ and that labelled by the variables in $\mathbf{x}_{d-1}$ and $\mathbf{x}_d$ is $B_{d-1,d}^{-1}$. For $k \in [3, d-2]$ the submatrix restricted to the rows and columns labelled by $\mathbf{x}_{\sigma(k)}$ is $B_{\sigma(k)}^{-1}$. We infer the following facts:

(I) The space spanned by the first $u_1 - u_2$ (that is w_1) rows of V^{-1} is equal to the space spanned by the first w_1 rows of A, the latter space is $\mathcal{X}_1$.

[41] If $\mathcal{V}_1$ and $\mathcal{V}_d$ are the spaces $A^{-1}\mathcal{U}_{d-1,d}$ and $A^{-1}\mathcal{U}_{1,2}$ respectively, then $\mathcal{Y}_1, \mathcal{Y}_{1,2}$ and $\mathcal{Y}_d, \mathcal{Y}_{d-1,d}$ are $\mathcal{X}_d, \mathcal{X}_{d-1,d}$ and $\mathcal{X}_1, \mathcal{X}_{1,2}$ respectively – the proof of this case is similar.

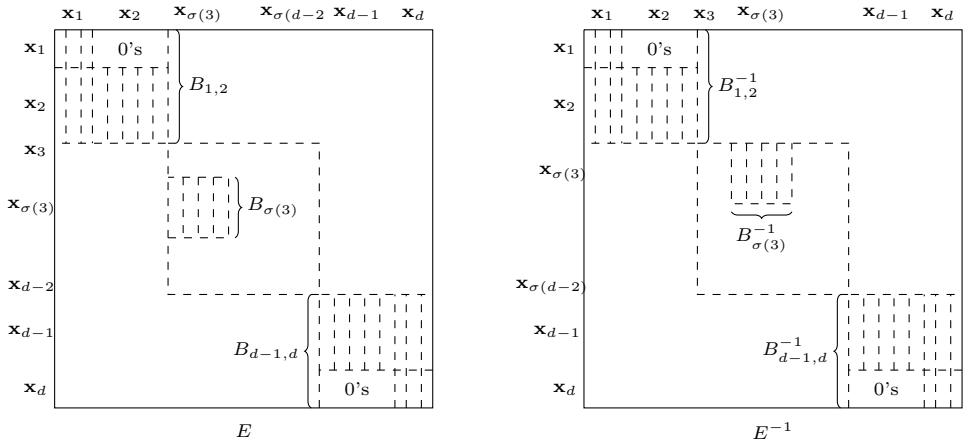


Figure 14 Matrices E and E^{-1}.

(II) The space spanned by the first u_1 (that is $w_1 + w_1w_2$) rows of V^{-1} is equal to the space spanned by the first $w_1 + w_1w_2$ rows of A, the latter space is $\mathcal{X}_{1,2}$.

(III) The space spanned by the last u_d (that is $w_{d-1} + w_{d-2}w_{d-1}$) rows of V^{-1} is equal to the space spanned by the last $w_{d-1} + w_{d-2}w_{d-1}$ rows of A, the latter space is $\mathcal{X}_{d-1,d}$.

(IV) The space spanned by the last $u_d - u_{d-1}$ (that is w_{d-1}) rows of V^{-1} is equal to the space spanned by the last w_{d-1} rows of A, the latter space is $\mathcal{X}_d$.

(V) For $k \in [3, d-2]$ the space spanned by the rows of V^{-1} that are numbered by $t_{k-1}+1$ to $t_{k-1}+u_k$ is equal to the space spanned by the rows of A labelled by $\mathbf{x}_{\sigma(k)}$, the latter space is $\mathcal{X}_{\sigma(k)}$. ◀

▶ **Claim 50** (restated). *There is a randomized polynomial time algorithm that takes input the bases of the layer spaces $\mathcal{Y}_1, \mathcal{Y}_{1,2}, \mathcal{Y}_3, \ldots, \mathcal{Y}_{d-2}, \mathcal{Y}_{d-1,d}, \mathcal{Y}_d$ and with probability at least $1 - \frac{1}{\text{poly}(n)}$ reorders these layer spaces and outputs a width vector $\mathbf{w}'$ such that the reordered sequence of spaces and $\mathbf{w}'$ are:*

1. *either $\mathcal{X}_1, \mathcal{X}_{1,2}, \mathcal{X}_3, \ldots, \mathcal{X}_{d-2}, \mathcal{X}_{d-1,d}, \mathcal{X}_d$ and $(w_1, w_2, \ldots, w_{d-1})$ respectively,*
2. *or $\mathcal{X}_d, \mathcal{X}_{d-1,d}, \mathcal{X}_{d-2}, \ldots, \mathcal{X}_3, \mathcal{X}_{1,2}, \mathcal{X}_1$ and $(w_d, w_{d-1}, \ldots, w_1)$ respectively.*

Proof. The algorithm employs evaluation dimension to uncover the permutation σ. Assume that $\mathcal{Y}_1, \mathcal{Y}_{1,2}, \mathcal{Y}_3, \ldots, \mathcal{Y}_{d-2}, \mathcal{Y}_{d-1,d}, \mathcal{Y}_d$ are the spaces $\mathcal{X}_1, \mathcal{X}_{1,2}, \mathcal{X}_{\sigma(3)}, \ldots, \mathcal{X}_{\sigma(d-2)}, \mathcal{X}_{d-1,d}, \mathcal{X}_d$ respectively[42]. In this case, the algorithm reorders the spaces to a sequence $\mathcal{X}_1, \mathcal{X}_{1,2}, \mathcal{X}_3, \ldots, \mathcal{X}_{d-2}, \mathcal{X}_{d-1,d}, \mathcal{X}_d$ and outputs $\mathbf{w}' = \mathbf{w}$. For every $k \in [3, d-2]$, let $\mathbf{z}_k$ be a set of $\dim(\mathcal{Y}_k)$ many variables. Let $\mathbf{z}_1$ (similarly, $\mathbf{z}_d$) be a set of $\dim(\mathcal{Y}_1)$ (similarly, $\dim(\mathcal{Y}_d)$) variables, and let $\mathbf{z}_2$ (similarly, $\mathbf{z}_{d-1}$) be a set of $\dim(\mathcal{Y}_{1,2}) - \dim(\mathcal{Y}_1)$ (similarly, $\dim(\mathcal{Y}_{d-1,d}) - \dim(\mathcal{Y}_d)$) variables. Finally, let $\mathbf{z} = \mathbf{z}_1 \uplus \ldots \uplus \mathbf{z}_d$ be the set of these n fresh variables.

Compute a linear map μ that maps $\mathbf{x}$ variables to linear forms in $\mathbf{z}$ variables such that the following conditions are satisfied:

(a) For every $k \in [3, d-2]$, the linear forms corresponding[43] to the basis vectors of $\mathcal{Y}_k$ map to distinct variables in $\mathbf{z}_k$.

(b) The linear forms corresponding to the basis vectors in $\mathcal{Y}_1$ (similarly, $\mathcal{Y}_d$) map to distinct variables in $\mathbf{z}_1$ (similarly, $\mathbf{z}_d$).

[42] The proof of the other case is similar.
[43] Recall, linear forms in $\mathbf{x}$ variables and vectors in $\mathbb{F}^n$ are naturally identified with each other.

(c) The linear forms corresponding to the basis vectors in $\mathcal{Y}_{1,2}$ (similarly, $\mathcal{Y}_{d-1,d}$) map to distinct variables in $\mathbf{z}_1 \uplus \mathbf{z}_2$ (similarly, $\mathbf{z}_{d-1} \uplus \mathbf{z}_d$).

Conditions (b) and (c) can be simultaneously satisfied as the basis of $\mathcal{Y}_1$ (similarly, $\mathcal{Y}_d$) is contained in the basis of $\mathcal{Y}_{1,2}$ (similarly, $\mathcal{Y}_{d-1,d}$) by their very constructions in Algorithm 5. As $f = \mathsf{IMM}_{\mathbf{w},d}(A\mathbf{x})$, the map μ takes f to a polynomial $h(\mathbf{z})$ that is computed by a full rank ABP $\mathtt{A}'$ of width $\mathbf{w}$ and length d such that the sets of variables appearing in the d layers of $\mathtt{A}'$ from left to right are $\mathbf{z}_1, \mathbf{z}_1 \uplus \mathbf{z}_2, \mathbf{z}_{\sigma^{-1}(3)}, \ldots, \mathbf{z}_{\sigma^{-1}(d-2)}, \mathbf{z}_{d-1} \uplus \mathbf{z}_d, \mathbf{z}_d$ in order.

The following observation, the proof of which is given later, helps find σ^{-1} incrementally from blackbox access to $h(\mathbf{z})$. Let $\mathbf{y}_2 = \mathbf{z}_1 \uplus \mathbf{z}_2$ and $\mathbf{y}_j = \mathbf{z}_1 \uplus \mathbf{z}_2 \uplus \mathbf{z}_{\sigma^{-1}(3)} \uplus \cdots \uplus \mathbf{z}_{\sigma^{-1}(j)}$, for $j \in [3, d-2]$.

▶ **Observation 67.** *For every $j \in [2, d-3]$ and $k \in [3, d-2]$ such that $\mathbf{z}_k \not\subset \mathbf{y}_j$,*

1. $\mathrm{Evaldim}_{\mathbf{y}_j \uplus \mathbf{z}_k}(h) < |\mathbf{z}_k|$, *if $k = \sigma^{-1}(j+1)$, and*
2. $\mathrm{Evaldim}_{\mathbf{y}_j \uplus \mathbf{z}_k}(h) > |\mathbf{z}_k|$, *if $k \neq \sigma^{-1}(j+1)$.*

The proof of the observation also includes an efficient randomized procedure to compute $\mathrm{Evaldim}_{\mathbf{y}_j \uplus \mathbf{z}_k}(h)$.

Finally, the algorithm outputs the reordered layer spaces $\mathcal{Y}_1, \mathcal{Y}_{1,2}, \mathcal{Y}_{\sigma^{-1}(3)}, \ldots, \mathcal{Y}_{\sigma^{-1}(d-2)}, \mathcal{Y}_{d-1,d}, \mathcal{Y}_d$ which is the ordered sequence of spaces $\mathcal{X}_1, \mathcal{X}_{1,2}, \mathcal{X}_3, \ldots, \mathcal{X}_{d-2}, \mathcal{X}_{d-1,d}, \mathcal{X}_d$. The width vector $\mathbf{w}'$ can be readily calculated now by inspecting the dimensions:

$$\begin{aligned}
w'_1 &= \dim(\mathcal{X}_1) = w_1, \\
w'_2 &= \frac{\dim(\mathcal{X}_{1,2}) - w_1}{w_1} = w_2, \\
w'_k &= \frac{\dim(\mathcal{X}_k)}{w_{k-1}} = w_k, \quad \text{for } k \in [3, d-2], \\
w'_d &= \dim(\mathcal{X}_d) = w_d, \quad \text{and} \\
w'_{d-1} &= \frac{\dim(\mathcal{X}_{d-1,d}) - w_d}{w_d} = w_{d-1}.
\end{aligned}$$

This gives $\mathbf{w}' = \mathbf{w}$. ◀

Proof of Observation 67. Let $Z_1 \cdot Z_2 \cdots Z_d$ be equal to $\mathtt{A}'$, the full rank ABP of width $\mathbf{w} = (w_1, w_2, \ldots, w_{d-1})$ computing h, where the linear forms in Z_i are in $\mathbf{z}_{\sigma^{-1}(i)}$ variables for $i \in [3, d-2]$, the linear forms in Z_1, Z_d are in variables $\mathbf{z}_1, \mathbf{z}_d$ respectively, and the linear forms in Z_2, Z_{d-1} are in $\mathbf{z}_1 \uplus \mathbf{z}_2, \mathbf{z}_{d-1} \uplus \mathbf{z}_d$ variables respectively.

Case 1: Suppose $k = \sigma^{-1}(j+1)$, implying $|\mathbf{z}_k| = w_j w_{j+1}$. Let $G = Z_{j+2} \cdot Z_{j+3} \cdots Z_d$ and the t-th entry of G be g_t for $t \in [w_{j+1}]$. As the linear forms in $Z_1, Z_2, \ldots, Z_{j+1}$ are $\mathbb{F}$-linearly independent, for every $t \in [w_{j+1}]$ there is a partial evaluation of h at $\mathbf{y}_j \uplus \mathbf{z}_k$ variables that makes h equal to g_t . Also, every partial evaluation of h at $\mathbf{y}_j \uplus \mathbf{z}_k$ variables can be expressed as an $\mathbb{F}$-linear combination of $g_1, g_2, \ldots, g_{w_{j+1}}$. Hence, from Observation 53 it follows, $\mathrm{Evaldim}_{\mathbf{y}_j \uplus \mathbf{z}_k}(h) = w_{j+1} < |\mathbf{z}_k|$.

Case 2: Suppose $k \neq \sigma^{-1}(j+1)$. The variables $\mathbf{z}_k$ appear in the matrix $Z_{\sigma(k)}$, so $|\mathbf{z}_k| = w_{\sigma(k)-1} w_{\sigma(k)}$. Let $G = Z_{\sigma(k)+1} \cdot Z_{\sigma(k)+2} \cdots Z_d$ and the t-th entry of G be g_t for $t \in [w_{\sigma(k)}]$. Further, let $P = (p_{lm})_{l \in [w_j], m \in [w_{\sigma(k)-1}]}$ be equal to $Z_{j+1} \cdot Z_{j+2} \cdots Z_{\sigma(k)-1}$. As the linear forms in $Z_1, Z_2, \ldots, Z_j$ and $Z_{\sigma(k)}$ are $\mathbb{F}$-linearly independent, there is a partial evaluation of h at the $\mathbf{y}_j \uplus \mathbf{z}_k$ variables that makes h equal to $p_{lm} g_t$ for $l \in [w_j], m \in [w_{\sigma(k)-1}]$ and $t \in [w_{\sigma(k)}]$. By Observation 53, $\{g_t \mid t \in [w_{\sigma(k)}]\}$ are $\mathbb{F}$-linearly independent; using a proof similar to that of Observation 53 we can show that the polynomials $\{p_{lm} \mid l \in$

$[w_j], m \in [w_{\sigma(k)-1}]\}$ are also $\mathbb{F}$-linearly independent. This implies the set of polynomials $\{p_{lm}g_t \mid l \in [w_j], m \in [w_{\sigma(k)-1}]$ and $t \in [w_{\sigma(k)}]\}$ are $\mathbb{F}$-linearly independent, as p_{lm} and g_t are on disjoint sets of variables. Since every partial evaluation of h at $\mathbf{y}_j \uplus \mathbf{z}_k$ variables can be expressed as an $\mathbb{F}$-linear combination of the set of polynomials $\{p_{lm}g_t \mid l \in [w_j], m \in [w_{\sigma(k)-1}]$ and $t \in [w_{\sigma(k)}]\}$, $\text{Evaldim}_{\mathbf{y}_j \uplus \mathbf{z}_k}(h) = w_j w_{\sigma(k)-1} w_{\sigma(k)} = w_j \cdot |\mathbf{z}_k| > |\mathbf{z}_k|$.

A randomized procedure to compute $\text{Evaldim}_{\mathbf{y}_j \uplus \mathbf{z}_k}(h)$: Choose evaluation points $\mathbf{a}_1, \ldots, \mathbf{a}_{n^2}$ for the variables $\mathbf{y}_j \uplus \mathbf{z}_k$ independently and uniformly at random from a set $S^{|\mathbf{y}_j \uplus \mathbf{z}_k|} \subset \mathbb{F}^{|\mathbf{y}_j \uplus \mathbf{z}_k|}$ with $|S| = \text{poly}(n)$. Output the dimension of the $\mathbb{F}$-linear space spanned by the polynomials $h(\mathbf{a}_1), \ldots, h(\mathbf{a}_{n^2})$ using Claim 24.

We argue that the above procedure outputs $\text{Evaldim}_{\mathbf{y}_j \uplus \mathbf{z}_k}(h)$ with probability at least $1 - \frac{1}{\text{poly}(n)}$. Let $\text{Evaldim}_{\mathbf{y}_j \uplus \mathbf{z}_k}(h) = e$. Observe that in both Case 1 and 2, $e \leq n^2$. Also, in both the cases h can be expressed as

$$h = \sum_{i \in [e]} f_i(\mathbf{y}_j \uplus \mathbf{z}_k) \cdot q_i, \tag{25}$$

where f_i and q_i are variable disjoint. The polynomials $q_1, \ldots, q_e$ are the polynomials $g_1, \ldots, g_{w_{j+1}}$ in Case 1; they are the polynomials $\{p_{lm}g_t \mid l \in [w_j], m \in [w_{\sigma(k)-1}]$ and $t \in [w_{\sigma(k)}]\}$ in Case 2. Just as we argue that $q_1, \ldots, q_e$ are $\mathbb{F}$-linearly independent, we can show that $f_1, \ldots, f_e$ are also $\mathbb{F}$-linearly independent. So, by Claim 24 the rank of the matrix $M = (f_c(\mathbf{a}_r))_{r,c \in [e]}$ is e with high probability. This implies the polynomials $h(\mathbf{a}_1), \ldots, h(\mathbf{a}_e)$ are $\mathbb{F}$-linearly independent also with high probability. The correctness of the procedure follows from the observation that the dimension of the $\mathbb{F}$-linear space spanned by $h(\mathbf{a}_1), \ldots, h(\mathbf{a}_{n^2})$ is upper bounded by e (from Equation (25)). ◀

▶ **Observation 52** (restated). *If h is computable by a full rank almost set-multilinear ABP of width $\mathbf{w}$ then there is a full rank almost set-multilinear ABP of width $\mathbf{w}$ in canonical form computing h.*

Proof. Suppose $X_1 \cdot X_2 \cdots X_d$ is a full rank almost set-multilinear ABP of width $\mathbf{w} = (w_1, w_2, \ldots, w_{d-1})$ computing h. Let $X'_1 = (x_1^{(1)}\ x_2^{(1)} \ldots\ x_{w_1}^{(1)})$ and $X'_d = (x_1^{(d)}\ x_2^{(d)} \ldots\ x_{w_{d-1}}^{(d)})$. We show there are matrices X'_2 and X'_{d-1} satisfying conditions (1b) and (2b) respectively of canonical form (defined in Section 2.4) such that $h = X'_1 \cdot X'_2 \cdot X_3 \cdots X_{d-2} \cdot X'_{d-1} \cdot X'_d$. We prove the existence of $X'_2 = (l'_{ij})_{i \in [w_1], j \in [w_2]}$; the proof for X'_{d-1} is similar. It is sufficient to show that there is such an X'_2 satisfying $X_1 \cdot X_2 = X'_1 \cdot X'_2$. Denote the j-th entry of the $1 \times w_2$ matrix $X_1 \cdot X_2$ as $X_1 \cdot X_2(j)$. Similarly $X'_1 \cdot X'_2(j)$ represents the j-th entry of $X'_1 \cdot X'_2$. Let g_i be the sum of all monomials in $X_1 \cdot X_2(j)$ of the following types: $x_i^{(1)} x_k^{(1)}$ for $k \in [i, w_1]$, and $x_i^{(1)} x_{pq}^{(2)}$ for $p \in [w_1], q \in [w_2]$. Clearly,

$$X_1 \cdot X_2(j) = g_1 + g_2 + \cdots + g_{w_1}.$$

If $l'_{ij} \stackrel{\text{def}}{=} g_i / x_i^{(1)}$ then

$$X_1 \cdot X_2(j) = x_1^{(1)} l'_{1j} + x_2^{(1)} l'_{2j} + \cdots + x_{w_1}^{(1)} l'_{w_1 j}.$$

Since l'_{ij} is the (i,j)-th entry of X'_2, we infer $X_1 \cdot X_2(j) = X'_1 \cdot X'_2(j)$. By definition, $x_k^{(1)}$ does not appear in l'_{ij} for $k < i$, and thus condition (1b) is satisfied by X'_2. ◀

▶ **Observation 53** (restated). *Let $X_1 \cdot X_2 \cdots X_d$ be a full rank almost set-multilinear ABP, and $C_k = X_k \cdots X_d$ for $k \in [2, d]$. Let the l-th entry of C_k be h_{kl} for $l \in [w_{k-1}]$. Then the polynomials $\{h_{k1}, h_{k2}, \cdots, h_{kw_{k-1}}\}$ are $\mathbb{F}$-linearly independent.*

Proof. Suppose $\sum_{p=1}^{w_{k-1}} \alpha_p \cdot h_{kp} = 0$ such that $\alpha_p \in \mathbb{F}$ for $p \in [w_{k-1}]$, and not all $\alpha_p = 0$. Assume without loss of generality $\alpha_1 \neq 0$. Since the linear forms in $X_k, \ldots, X_d$ are $\mathbb{F}$-linearly independent, there is an evaluation of the variables in $\mathbf{x}_k \uplus \cdots \uplus \mathbf{x}_d$ to field constants such that $h_{k1} = 1$ and every other $h_{kp} = 0$ under this evaluation. This implies $\alpha_1 = 0$, contradicting our assumption. ◀

7.5 Proof of observation in Section 6

▶ **Observation 57** (restated). *There are matrices $A_1, \ldots, A_{d-1}$ with $A_k \in \mathsf{GL}(w_k)$ for every $k \in [d-1]$, such that $X_1 = Q_1 \cdot A_1$, $X_2(\mathbf{x}_2) = A_1^{-1} \cdot Q_2 \cdot A_2$, $X_{d-1}(\mathbf{x}_{d-1}) = A_{d-2}^{-1} \cdot Q_{d-1} \cdot A_{d-1}$, $X_d = A_{d-1}^{-1} \cdot Q_d$, and $X_k = A_{k-1}^{-1} \cdot Q_k \cdot A_k$ for $k \in [3, d-2]$.*

Proof. To simplify notations, we write $X_2(\mathbf{x}_2)$, $X_{d-1}(\mathbf{x}_{d-1})$ as X_2, X_{d-1} respectively. We have

$$X_1 \cdot X_2 \cdots X_{d-1} \cdot X_d \;=\; Q_1 \cdot Q_2 \cdots Q_{d-1} \cdot Q_d \;=\; \mathsf{IMM},$$

where the dimensions of the matrices X_k and Q_k are the same, and the set of variables appearing in both X_k and Q_k is $\mathbf{x}_k$, for every $k \in [d]$. Since the linear forms in X_1 are $\mathbb{F}$-linearly independent, there is an $A_1 \in \mathsf{GL}(w_1)$ such that $X_1 = Q_1 \cdot A_1$, implying

$$\begin{aligned} & Q_1 \cdot [A_1 \cdot X_2 \cdots X_{d-1} \cdot X_d \;-\; Q_2 \cdots Q_{d-1} \cdot Q_d] \;=\; 0 \\ \Rightarrow\; & X_2 \cdots X_{d-1} \cdot X_d \;=\; A_1^{-1} \cdot Q_2 \cdots Q_{d-1} \cdot Q_d, \end{aligned}$$

as the formal variable entries of Q_1 do not appear in the matrices X_k, Q_k for $k \in [2, d]$. The rest of the proof proceeds inductively: Suppose for some $k \in [2, d-1]$,

$$X_k \cdots X_{d-1} \cdot X_d \;=\; A_{k-1}^{-1} \cdot Q_k \cdots Q_{d-1} \cdot Q_d, \quad \text{where } A_{k-1} \in \mathsf{GL}(w_{k-1}).$$

Let $p_k = \sum_{i=k+1}^{d} |\mathbf{x}_i|$. Since the linear forms in $X_{k+1}, \ldots, X_{d-1}, X_d$ are $\mathbb{F}$-linearly independent, for every $l \in [w_k]$ there is a point $\mathbf{a}_l \in \mathbb{F}^{p_k}$ such that the $w_k \times 1$ matrix $X_{k+1} \cdots X_{d-1} \cdot X_d$ evaluated at $\mathbf{a}_l$ has 1 at the l-th position and all its other entries are zero. Let A_k be the $w_k \times w_k$ matrix such that the l-th column of A_k is equal to $Q_{k+1} \cdots Q_{d-1} \cdot Q_d$ evaluated at $\mathbf{a}_l$. Then, $X_k = A_{k-1}^{-1} \cdot Q_k \cdot A_k$. As the linear forms in X_k and Q_k are $\mathbb{F}$-linearly independent, it must be that $A_k \in \mathsf{GL}(w_k)$. Putting this expression for X_k in the equation above and arguing as before, we get a similar equation with k replaced by $k+1$. The proof then follows by induction. ◀

References

1. Scott Aaronson. Arithmetic natural proofs theory is sought. `http://www.scottaaronson.com/blog/?p=336`, 2008.
2. Manindra Agrawal. Proving Lower Bounds Via Pseudo-random Generators. In *FSTTCS 2005: Foundations of Software Technology and Theoretical Computer Science, 25th International Conference, Hyderabad, India, December 15-18, 2005, Proceedings*, pages 92–105, 2005.
3. Manindra Agrawal, Rohit Gurjar, Arpita Korwar, and Nitin Saxena. Hitting-Sets for RO-ABP and Sum of Set-Multilinear Circuits. *SIAM J. Comput.*, 44(3):669–697, 2015.
4. Manindra Agrawal and Nitin Saxena. Equivalence of f-algebras and cubic forms. In *STACS 2006, 23rd Annual Symposium on Theoretical Aspects of Computer Science, Marseille, France, February 23-25, 2006, Proceedings*, pages 115–126, 2006.
5. Dana Angluin. Queries and concept learning. *Machine Learning.*, 2(4):319–342, 1988.

6 Vikraman Arvind, Partha Mukhopadhyay, and Srikanth Srinivasan. New Results on Noncommutative and Commutative Polynomial Identity Testing. In *Proceedings of the 23rd Annual IEEE Conference on Computational Complexity, CCC 2008, 23-26 June 2008, College Park, Maryland, USA*, pages 268–279, 2008.

7 Amos Beimel, Francesco Bergadano, Nader H. Bshouty, Eyal Kushilevitz, and Stefano Varricchio. Learning functions represented as multiplicity automata. *J. ACM*, 47(3):506–530, 2000.

8 Michael Ben-Or and Richard Cleve. Computing algebraic formulas using a constant number of registers. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 254–257, 1988.

9 Elwyn Berlekamp. Factoring polynomials over finite fields. *Bell System Technical Journal*, 46:1853–1859, 1967.

10 Lenore Blum, Mike Shub, and Steve Smale. On a Theory of Computation over the Real Numbers; NP Completeness, Recursive Functions and Universal Machines (Extended Abstract). In *29th Annual Symposium on Foundations of Computer Science, White Plains, New York, USA, 24-26 October 1988*, pages 387–397, 1988.

11 David G. Cantor and Hans Zassenhaus. A new algorithm for factoring polynomials over finite fields. *Mathematics of Computation*, 36:587–592, 1981.

12 Enrico Carlini. Reducing the number of variables of a polynomial. In *Algebraic geometry and geometric modelling, Mathematics and Visualization, Springer*, pages 237–247, 2006.

13 Zeev Dvir, Rafael Mendes de Oliveira, and Amir Shpilka. Testing equivalence of polynomials under shifts. In *Automata, Languages, and Programming – 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part I*, pages 417–428, 2014.

14 Michael A. Forbes and Amir Shpilka. Quasipolynomial-time identity testing of noncommutative and read-once oblivious algebraic branching programs. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 243–252, 2013.

15 Fulvio Gesmundo. Gemetric aspects of iterated matrix multiplication. *Journal of Algebra*, 461:42–64, 2016.

16 Joshua A. Grochow. *Symmetry and equivalence relations in classical and geometric complexity theory*. PhD thesis, The University of Chicago, 2012.

17 Ankit Gupta, Neeraj Kayal, and Satyanarayana V. Lokam. Efficient Reconstruction of Random Multilinear Formulas. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 778–787, 2011.

18 Ankit Gupta, Neeraj Kayal, and Satyanarayana V. Lokam. Reconstruction of depth-4 multilinear circuits with top fan-in 2. In *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19-22, 2012*, pages 625–642, 2012.

19 Ankit Gupta, Neeraj Kayal, and Youming Qiao. Random Arithmetic Formulas Can Be Reconstructed Efficiently. In *Proceedings of the 28th Conference on Computational Complexity, CCC 2013, K.lo Alto, California, USA, 5-7 June, 2013*, pages 1–9, 2013.

20 Johan Håstad. Tensor Rank is NP-Complete. In *Automata, Languages and Programming, 16th International Colloquium, ICALP89, Stresa, Italy, July 11-15, 1989, Proceedings*, pages 451–460, 1989.

21 Joos Heintz and Claus-Peter Schnorr. Testing Polynomials which Are Easy to Compute (Extended Abstract). In *Proceedings of the 12th Annual ACM Symposium on Theory of Computing, April 28-30, 1980, Los Angeles, California, USA*, pages 262–272, 1980.

22 Erich Kaltofen and Barry M. Trager. Computing with Polynomials Given By Black Boxes for Their Evaluation: Greatest Common Divisors, Factorization, Separation of Numerators and Denominators. In *29th Annual Symposium on Foundations of Computer Science, White Plains, New York, USA, 24-26 October 1988*, pages 296–305, 1988.

23 Zohar Shay Karnin and Amir Shpilka. Reconstruction of generalized depth-3 arithmetic circuits with bounded top fan-in. In *Proceedings of the 24th Annual IEEE Conference on Computational Complexity, CCC 2009, Paris, France, 15-18 July 2009*, pages 274–285, 2009.

24 Neeraj Kayal. Efficient algorithms for some special cases of the polynomial equivalence problem. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, San Francisco, California, USA, January 23-25, 2011*, pages 1409–1421, 2011.

25 Neeraj Kayal. Affine projections of polynomials: extended abstract. In *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19-22, 2012*, pages 643–662, 2012.

26 Neeraj Kayal. An exponential lower bound for the sum of powers of bounded degree polynomials. *Electronic Colloquium on Computational Complexity (ECCC)*, 19:81, 2012.

27 Neerak Kayal, Chandan Saha, and Sébastien Tavenas. An Average-Case Matrix Factorization Problem. Work in progress, 2017.

28 Adam Klivans, Pravesh Kothari, and Igor Carboni Oliveira. Constructing Hard Functions Using Learning Algorithms. In *Proceedings of the 28th Conference on Computational Complexity, CCC 2013, K.lo Alto, California, USA, 5-7 June, 2013*, pages 86–97, 2013.

29 Adam Klivans and Amir Shpilka. Learning arithmetic circuits via partial derivatives. 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 463–476, 2003.

30 Adam Klivans and Daniel A. Spielman. Randomness efficient identity testing of multivariate polynomials. In *Proceedings on 33rd Annual ACM Symposium on Theory of Computing, July 6-8, 2001, Heraklion, Crete, Greece*, pages 216–223, 2001.

31 A.K. Lenstra, H.W.jun. Lenstra, and László Lovász. Factoring polynomials with rational coefficients. *Math. Ann.*, 261:515–534, 1982.

32 Meena Mahajan and V. Vinay. Determinant: Combinatorics, algorithms, and complexity. *Chicago J. Theor. Comput. Sci.*, 1997, 1997.

33 Daniel Minahan and Ilya Volkovich. Complete Derandomization of Identity Testing and Reconstruction of Read-Once Formulas. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:171, 2016.

34 Noam Nisan. Lower bounds for non-commutative computation (extended abstract). In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing, May 5-8, 1991, New Orleans, Louisiana, USA*, pages 410–418, 1991.

35 Jacques Patarin. Hidden fields equations (HFE) and isomorphisms of polynomials (IP): two new families of asymmetric algorithms. In *Advances in Cryptology – EUROCRYPT'96, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding*, pages 33–48, 1996.

36 Alexander A. Razborov and Steven Rudich. Natural proofs. In *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, 23-25 May 1994, Montréal, Québec, Canada*, pages 204–213, 1994.

37 Amir Shpilka. Interpolation of depth-3 arithmetic circuits with two multiplication gates. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, June 11-13, 2007*, pages 284–293, 2007.

38 Amir Shpilka and Ilya Volkovich. Improved Polynomial Identity Testing for Read-Once Formulas. 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 700–713, 2009.

39 Amir Shpilka and Amir Yehudayoff. Arithmetic circuits: A survey of recent results and open questions. *Foundations and Trends in Theoretical Computer Science*, 5(3-4):207–388, 2010.

40 Gaurav Sinha. Reconstruction of real depth-3 circuits with top fan-in 2. In *31st Conference on Computational Complexity, CCC 2016, May 29 to June 1, 2016, Tokyo, Japan*, pages 31:1–31:53, 2016.

41 Thomas Thierauf. The isomorphism problem for read-once branching programs and arithmetic circuits. *Chicago J. Theor. Comput. Sci.*, 1998, 1998.

42 Ilya Volkovich. A guide to learning arithmetic circuits. In *Proceedings of the 29th Conference on Learning Theory, COLT 2016, New York, USA, June 23-26, 2016*, pages 1540–1561, 2016.

Complexity-Theoretic Foundations of Quantum Supremacy Experiments*

Scott Aaronson[1] and Lijie Chen[2]

1 **The University of Texas at Austin, Austin, TX, USA**
aaronson@cs.utexas.edu
2 **Tsinghua University, Beijing, China**
wjmzbmr@gmail.com

Abstract

In the near future, there will likely be special-purpose quantum computers with 40–50 high-quality qubits. This paper lays general theoretical foundations for how to use such devices to demonstrate "quantum supremacy": that is, a clear quantum speedup for *some* task, motivated by the goal of overturning the Extended Church-Turing Thesis as confidently as possible.

First, we study the hardness of sampling the output distribution of a random quantum circuit, along the lines of a recent proposal by the Quantum AI group at Google. We show that there's a natural average-case hardness assumption, which has nothing to do with sampling, yet implies that no polynomial-time classical algorithm can pass a statistical test that the quantum sampling procedure's outputs do pass. Compared to previous work – for example, on BosonSampling and IQP – the central advantage is that we can now talk directly about the observed outputs, rather than about the distribution being sampled.

Second, in an attempt to refute our hardness assumption, we give a new algorithm, inspired by Savitch's Theorem, for simulating a general quantum circuit with n qubits and depth d in polynomial space and $d^{O(n)}$ time. We then discuss why this and other known algorithms fail to refute our assumption.

Third, resolving an open problem of Aaronson and Arkhipov, we show that any strong quantum supremacy theorem – of the form "if approximate quantum sampling is classically easy, then the polynomial hierarchy collapses"– must be non-relativizing. This sharply contrasts with the situation for exact sampling.

Fourth, refuting a conjecture by Aaronson and Ambainis, we show that there is a sampling task, namely Fourier Sampling, with a *1 versus linear* separation between its quantum and classical query complexities.

Fifth, in search of a "happy medium"between black-box and non-black-box arguments, we study quantum supremacy relative to oracles in P/poly. Previous work implies that, if one-way functions exist, then quantum supremacy is possible relative to such oracles. We show, conversely, that *some* computational assumption is needed: if SampBPP = SampBQP and NP ⊆ BPP, then quantum supremacy is impossible relative to oracles with small circuits.

1998 ACM Subject Classification F.1.1 Models of Computation

Keywords and phrases computational complexity, quantum computing, quantum supremacy

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.22

* Scott Aaronson is supported by a Vannevar Bush Faculty Fellowship from the US Department of Defense, and by the Simons Foundation "It from Qubit" Collaboration. Lijie Chen is supported in part by the National Basic Research Program of China Grant 2011CBA00300, 2011CBA00301, the National Natural Science Foundation of China Grant 61361136003.

32nd Computational Complexity Conference (CCC 2017).
Editor: Ryan O'Donnell; Article No. 22; pp. 22:1–22:67

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

1 Introduction

The *Extended Church-Turing Thesis*, or ECT, asserts that every physical process can be simulated by a deterministic or probabilistic Turing machine with at most polynomial overhead. Since the 1980s – and certainly since the discovery of Shor's algorithm [55] in the 1990s – computer scientists have understood that quantum mechanics might refute the ECT in principle. Today, there are actual experiments being planned (e.g., [19]) with the goal of severely challenging the ECT in practice. These experiments don't yet aim to build full, fault-tolerant, universal quantum computers, but "merely" to demonstrate *some* quantum speedup over the best known or conjectured classical algorithms, for some possibly-contrived task, as confidently as possible. In other words, the goal is to answer the skeptics [41, 38] who claim that genuine quantum speedups are either impossible in theory, or at any rate, are hopelessly out of reach technologically. Recently, the term "quantum supremacy" has come into vogue for such experiments, footnoteAs far as we know, the first person to use the term in print was John Preskill [47]. although the basic goal goes back several decades, to the beginning of quantum computing itself.

Before going further, we should address some common misunderstandings about quantum supremacy.

The ECT is an asymptotic claim, which of course means that *no* finite experiment could render a decisive verdict on it, even in principle. But this hardly makes experiments irrelevant. If

1. a quantum device performed some task (say) 10^{15} times faster than a highly optimized simulation written by "adversaries" and running on a classical computing cluster, with the quantum/classical gap appearing to increase exponentially with the instance size across the whole range tested, and
2. this observed performance closely matched theoretical results that *predicted* such an exponential quantum speedup for the task in question, and
3. all other consistency checks passed (for example: removing quantum behavior from the experimental device destroyed the observed speedup),

this would obviously "raise the stakes" for anyone who still believed the ECT! Indeed, when some quantum computing researchers have criticized previous claims to have experimentally achieved quantum speedups (see, e.g., [2]), it has typically been on the ground that, in those researchers' view, the experiments failed to meet one or more of the conditions above.

It's sometimes claimed that any molecule in Nature or the laboratory, for which chemists find it computationally prohibitive to solve the Schrödinger equation and calculate its ground state, *already* provides an example of "quantum supremacy."The idea, in other words, is that such a molecule constitutes a "useful quantum computer, for the task of simulating itself."

For us, the central problem with this idea is that in theoretical computer science, we care less about individual *instances* than about solving *problems* (i.e., infinite collections of instances) in a more-or-less uniform way. For any one molecule, the difficulty in simulating it classically *might* reflect genuine asymptotic hardness, but it might also reflect other issues (e.g., a failure to exploit special structure in the molecule, or the same issues of modeling error, constant-factor overheads, and so forth that arise even in simulations of classical physics).

Thus, while it's possible that complex molecules could form the basis for a convincing quantum supremacy demonstration, we believe more work would need to be done. In particular, one would want a device that could synthesize *any* molecule in some theoretically infinite class – and one would then want complexity-theoretic evidence that the general

problem, of simulating a given molecule from that class, is asymptotically hard for a classical computer. And in such a case, it would seem more natural to call the synthesis machine the "quantum computer, textquotedblright rather than the molecules themselves!

In summary, we regard quantum supremacy as a central milestone for quantum computing that hasn't been reached yet, but that might be reached in the near future. This milestone is essentially negative in character: it has no obvious signature of the sort familiar to experimental physics, since it simply amounts to the *nonexistence* of an efficient classical algorithm to simulate a given quantum process. For that reason, the tools of theoretical computer science will be essential to understand when quantum supremacy has or hasn't been achieved. So in our view, even if it were uninteresting as TCS, there would still be an urgent need for TCS to contribute to the discussion about which quantum supremacy experiments to do, how to verify their results, and what should count as convincing evidence that classical simulation is hard. Happily, it turns out that there *is* a great deal here of intrinsic TCS interest as well.

1.1 Supremacy from Sampling

In recent years, a realization has crystallized that, if our goal is to demonstrate quantum supremacy (rather than doing anything directly useful), then there are good reasons to shift our attention from decision and function problems to *sampling* problems: that is, problems where the goal is to sample an n-bit string, either exactly or approximately, from a desired probability distribution.

A first reason for this is that demonstrating quantum supremacy via a sampling problem doesn't appear to require the full strength of a universal quantum computer. Indeed, there are now at least a half-dozen proposals [57, 23, 3, 44, 37, 29, 11] for special-purpose devices that could efficiently solve sampling problems believed to be classically intractable, *without* being able to solve every problem in the class BQP, or for that matter even every problem in P. Besides their intrinsic physical and mathematical interest, these intermediate models might be easier to realize than a universal quantum computer. In particular, because of their simplicity, they might let us avoid the need for the full machinery of quantum fault-tolerance [12]: something that adds a theoretically polylogarithmic but enormous-in-practice overhead to quantum computation. Thus, many researchers now expect that the first convincing demonstration of quantum supremacy will come via this route.

A second reason to focus on sampling problems is more theoretical: in the present state of complexity theory, we can arguably be *more* confident that certain quantum sampling problems really are classically hard, than we are that *factoring* (for example) is classically hard, or even that $\mathsf{BPP} \neq \mathsf{BQP}$. Already in 2002, Terhal and DiVincenzo [57] noticed that, while constant-depth quantum circuits can't solve any classically intractable decision problems, footnoteThis is because any qubit output by such a circuit depends on at most a constant number of input qubits. they nevertheless have a curious power: namely, they can sample probability distributions that can't be sampled in classical polynomial time, unless $\mathsf{BQP} \subseteq \mathsf{AM}$, which would be a surprising inclusion of complexity classes. Then, in 2004, Aaronson showed that $\mathsf{PostBQP} = \mathsf{PP}$, where $\mathsf{PostBQP}$ means BQP with the ability to postselect on exponentially-unlikely measurement outcomes. This had the immediate corollary that, if there's an efficient classical algorithm to sample the output distribution of an arbitrary quantum circuit – or for that matter, any distribution whose probabilities are multiplicatively close to the correct ones – then

$$\mathsf{PP} = \mathsf{PostBQP} = \mathsf{PostBPP} \subseteq \mathsf{BPP}^{\mathsf{NP}}.$$

By Toda's Theorem [58], this implies that the polynomial hierarchy collapses to the third level.

Related to that, in 2009, Aaronson [1] showed that, while it was (and remains) a notorious open problem to construct an oracle relative to which $\mathsf{BQP} \not\subset \mathsf{PH}$, one can construct oracular *sampling* and *relation* problems that are solvable in quantum polynomial time, but that are provably not solvable in randomized polynomial time augmented with a PH oracle.

Then, partly inspired by that oracle separation, Aaronson and Arkhipov [7, 3] proposed BosonSampling: a model that uses identical photons traveling through a network of beamsplitters and phaseshifters to solve classically hard sampling problems. Aaronson and Arkhipov proved that a polynomial-time exact classical simulation of BosonSampling would collapse PH. They also gave a plausible conjecture implying that even an *approximate* simulation would have the same consequence. Around the same time, Bremner, Jozsa, and Shepherd [23] independently proposed the Commuting Hamiltonians or IQP ("Instantaneous Quantum Polynomial-Time") model, and showed that it had the same property, that exact classical simulation would collapse PH. Later, Bremner, Montanaro, and Shepherd [24, 25] showed that, just like for BosonSampling, there are plausible conjectures under which even a fast classical *approximate* simulation of the IQP model would collapse PH.

Since then, other models have been proposed with similar behavior. To take a few examples: Farhi and Harrow [29] showed that the so-called Quantum Approximate Optimization Algorithm, or QAOA, can sample distributions that are classically intractable unless PH collapses. Morimae, Fujii, and Fitzsimons [44] showed the same for the so-called One Clean Qubit or DQC1 model, while Jozsa and Van den Nest [37] showed it for stabilizer circuits with magic initial states and nonadaptive measurements, and Aaronson et al. [11] showed it for a model based on integrable particle scattering in $1+1$ dimensions. In retrospect, the constant-depth quantum circuits considered by Terhal and DiVincenzo [57] also have the property that fast exact classical simulation would collapse PH.

Within the last four years, quantum supremacy via sampling has made the leap from complexity theory to a serious experimental prospect. For example, there have by now been many small-scale demonstrations of BosonSampling in linear-optical systems, with the current record being a 6-photon experiment by Carolan et al. [27]. To scale up to (say) 30 or 40 photons – as would be needed to make a classical simulation of the experiment suitably difficult – seems to require more reliable single-photon sources than exist today. But some experts (e.g., [50, 46]) are optimistic that optical multiplexing, superconducting resonators, or other technologies currently under development will lead to such photon sources. In the meantime, as we mentioned earlier, Boixo et al. [19] have publicly detailed a plan, currently underway at Google, to perform a quantum supremacy experiment involving random circuits applied to a 2D array of 40-50 coupled superconducting qubits. So far, the group at Google has demonstrated the preparation and measurement of entangled states on a linear array of 9 superconducting qubits [39].

1.2 Theoretical Challenges

Despite the exciting recent progress in both theory and experiment, some huge conceptual problems have remained about sampling-based quantum supremacy. These problems are not specific to any one quantum supremacy proposal (such as BosonSampling, IQP, or random quantum circuits), but apply with minor variations to all of them.

Verification of Quantum Supremacy Experiments. From the beginning, there was the problem of *how to verify the results* of a sampling-based quantum supremacy experiment. In contrast to (say) factoring and discrete log, for sampling tasks such as BosonSampling, it seems unlikely that there's any NP witness certifying the quantum experiment's output, let alone an NP witness that's also the experimental output itself. Rather, for the sampling tasks, not only simulation but even verification might need classical exponential time. Yet, while no one has yet discovered a general way around this, footnoteIn principle, one could use so-called *authenticated quantum computing* [13, 26], but the known schemes for that might be much harder to realize technologically than a basic quantum supremacy experiment, and in any case, they all presuppose the validity of quantum mechanics. it's far from the fatal problem that some have imagined. The reason is simply that experiments can and will target a "sweet spot, textquotedblright of (say) 40–50 qubits, for which classical simulation and verification of the results is *difficult but not impossible*.

Still, the existing verification methods have a second drawback. Namely, once we've fixed a specific verification test for sampling from a probability distribution $\mathcal{D}$, we ought to consider, not merely all classical algorithms that sample exactly or approximately from $\mathcal{D}$, but *all classical algorithms that output anything that passes the verification test.* To put it differently, we ought to talk not about the sampling problem itself, but about an associated *relation problem*: that is, a problem where the goal is to produce any output that satisfies a given condition.

As it happens, in 2011, Aaronson [8] proved an extremely general connection between sampling problems and relation problems. Namely, given any approximate sampling problem S, he showed how to define a relation problem R_S such that, for every "reasonable" model of computation (classical, quantum, etc.), R_S is efficiently solvable in that model if and only if S is. This had the corollary that

$$\mathsf{SampBPP} = \mathsf{SampBQP} \iff \mathsf{FBPP} = \mathsf{FBQP},$$

where SampBPP and SampBQP are the classes of approximate sampling problems solvable in polynomial time by randomized and quantum algorithms respectively, and FBPP and FBQP are the corresponding classes of relation problems. Unfortunately, Aaronson's construction of R_S involved Kolmogorov complexity: basically, one asks for an m-tuple of strings, $\langle x_1, \ldots, x_m \rangle$, such that

$$K(x_1, \ldots, x_m) \geq \log_2 \frac{1}{p_1 \cdots p_m} - O(1),$$

where p_i is the desired probability of outputting x_i in the sampling problem. And of course, verifying such a condition is extraordinarily difficult, even more so than calculating the probabilities $p_1, \ldots, p_m$.[1] For this reason, it's strongly preferable to have a condition that talks only about the largeness of the p_i's, and not about the algorithmic randomness of the x_i's. But then hardness for the sampling problem no longer necessarily implies hardness for the relation problem, so a new argument is needed.

Supremacy Theorems for Approximate Sampling. A second difficulty is that any quantum sampling device is subject to noise and decoherence. Ultimately, of course, we'd like hardness

[1] Furthermore, this is true even if we substitute a resource-bounded Kolmogorov complexity, as Aaronson's result allows.

results for quantum sampling that apply even in the presence of experimentally realistic errors. Very recently, Bremner, Montanaro, and Shepherd [25] and Fujii [31] have taken some promising initial steps in that direction. But even if we care only about the *smallest* "experimentally reasonable" error – namely, an error that corrupts the output distribution $\mathcal{D}$ to some other distribution $\mathcal{D}'$ that's ε-close to $\mathcal{D}$ in variation distance – Aaronson and Arkhipov [3] found that we already engage substantial new open problems in complexity theory, if we want evidence for classical hardness. So for example, their hardness argument for approximate BosonSampling depended on the conjecture that there's no $\mathsf{BPP}^{\mathsf{NP}}$ algorithm to estimate the permanent of an i.i.d. Gaussian matrix $A \sim N(0,1)_{\mathbb{C}}^{n\times n}$, with high probability over the choice of A.

Of course, one could try to close that loophole by proving that this Gaussian permanent estimation problem is #P-hard, which is indeed a major challenge that Aaronson and Arkhipov left open. But this situation also raises more general questions. For example, is there an implication of the form "if SampBPP = SampBQP, then PH collapses, textquotedblright where again SampBPP and SampBQP are the *approximate* sampling versions of BPP and BQP respectively? Are there oracles relative to which such an implication does *not* hold?

Quantum Supremacy Relative to Oracles. A third problem goes to perhaps the core issue of complexity theory (both quantum and classical): namely, we don't at present have a proof of P $\neq$ PSPACE, much less of BPP $\neq$ BQP or SampBPP $\neq$ SampBQP, less still of the hardness of specific problems like factoring or estimating Gaussian permanents. So what reason do we have to believe that *any* of these problems are hard? Part of the evidence has always come from oracle results, which we often *can* prove unconditionally. Particularly in quantum complexity theory, oracle separations can already be highly nontrivial, and give us a deep intuition for why all the "standard"algorithmic approaches fail for some problem.

On the other hand, we also know, from results like IP = PSPACE [54], that oracle separations can badly mislead us about what happens in the unrelativized world. Generally speaking, we might say, relying on an oracle separation is more dangerous, the less the oracle function resembles what would actually be available in an explicit problem.[2]

In the case of sampling-based quantum supremacy, we've known strong oracle separations since early in the subject. Indeed, in 2009, Aaronson [1] showed that Fourier Sampling – a quantumly easy sampling problem that involves only a *random* oracle – requires classical exponential time, and for that matter, sits outside the entire polynomial hierarchy. But of course, in real life random oracles are unavailable. So a question arises: can we say anything about the classical hardness of Fourier Sampling with a *pseudorandom* oracle? More broadly, what hardness results can we prove for quantum sampling, relative to oracles that are efficiently computable? Here, we imagine that an algorithm doesn't have access to a succinct representation of the oracle function f, but it does know that a succinct representation *exists* (i.e., that $f \in \mathsf{P/poly}$). Under that assumption, is there any hope of proving an *unconditional* separation between quantum and classical sampling? If not, then can we at least prove quantum supremacy under weaker (or more "generic") assumptions than would be needed in the purely computational setting?

[2] Indeed, the *algebrization barrier* of Aaronson and Wigderson [6] was based on precisely this insight: namely, if we force oracles to be "more realistic, textquotedblright by demanding (in that case) that they come equipped with algebraic extensions of whichever Boolean functions they represent, then many previously non-relativizing results become relativizing.

1.3 Our Contributions

In this paper, we address all three of the above challenges. Our results might look wide-ranging, but they're held together by a single thread: namely, *the quest to understand the classical hardness of quantum approximate sampling problems, and especially the meta-question of under which computational assumptions such hardness can be proven.* We'll be interested in both "positive" results, of the form "quantum sampling problem X is classically hard under assumption Y, textquotedblright and "negative" results, of the form "proving the classical hardness of X *requires* assumption Y."Also, we'll be less concerned with specific proposals such as BosonSampling, than simply with the general task of approximately sampling the output distribution of a given quantum circuit C. Fortuitously, though, our focus on quantum circuit sampling will make some of our results an excellent fit to currently planned experiments – most notably, those at Google [19], which will involve random quantum circuits on a 2D square lattice of 40 to 50 superconducting qubits. Even though we won't address the details of those or other experiments, our results (together with other recent work [19, 25]) can help to inform the experiments – for example, by showing how the circuit depth, the verification test applied to the outputs, and other design choices affect the strength of the computational assumptions that are necessary and sufficient to conclude that quantum supremacy has been achieved.

We have five main results.

The Hardness of Quantum Circuit Sampling. Our first result, in Section 3, is about the hardness of sampling the output distribution of a random quantum circuit, along the general lines of the planned Google experiment. Specifically, we propose a simple verification test to apply to the outputs of a random quantum circuit. We then analyze the classical hardness of generating *any* outputs that pass that test.

More concretely, we study the following basic problem:

▶ **Problem 1** (HOG, or Heavy Output Generation). *Given as input a random quantum circuit C (drawn from some suitable ensemble), generate output strings $x_1, \ldots, x_k$, at least a $2/3$ fraction of which have greater than the median probability in C's output distribution.*

HOG is a relation problem, for which we can verify a claimed solution in classical exponential time, by calculating the ideal probabilities $p_{x_1}, \ldots, p_{x_k}$ for each x_i to be generated by C, and then checking whether enough of the p_{x_i}'s are greater than the median value (which we can estimate analytically to extremely high confidence). Furthermore, HOG is easy to solve on a quantum computer, with overwhelming success probability, by the obvious strategy of just running C over and over and collecting k of its outputs.[3]

It certainly seems plausible that HOG is exponentially hard for a classical computer. But we ask: under what assumption could that hardness be proven? To address that question, we propose a new hardness assumption:

▶ **Assumption 1** (QUATH, or the QUAntum THreshold assumption). *There is no polynomial-time classical algorithm that takes as input a description of a random quantum circuit C, and that guesses whether $|\langle 0^n|C|0^n\rangle|^2$ is greater or less than the median of all 2^n of the $|\langle 0^n|C|x\rangle|^2$ values, with success probability at least $\frac{1}{2} + \Omega\left(\frac{1}{2^n}\right)$ over the choice of C.*

[3] Heuristically, one expects the p_{x_i}'s to be exponentially distributed random variables, which one can calculate implies that a roughly $\frac{1+\ln 2}{2} \approx 0.847$ fraction of the outputs will have probabilities exceeding the median value.

Our first result says that if QUATH is true, then HOG is hard. While this might seem nearly tautological, the important point here is that QUATH makes no reference to sampling or relation problems. Thus, we can now shift our focus from sampling algorithms to algorithms that simply estimate amplitudes, with a minuscule advantage over random guessing.

New Algorithms to Simulate Quantum Circuits. But given what a tiny advantage $\Omega\left(2^{-n}\right)$ is, why would anyone even conjecture that QUATH might be true? This brings us to our second result, in Section 4, which is motivated by the attempt to refute QUATH. We ask: what *are* the best classical algorithms to simulate an arbitrary quantum circuit? For special quantum circuits (e.g., those with mostly Clifford gates and a few T gates [22]), there's been exciting recent progress on improved exponential-time simulation algorithms, but for arbitrary quantum circuits, one might think there isn't much to say. Nevertheless, we *do* find something basic to say that, to our knowledge, had been overlooked earlier.

For a quantum circuit with n qubits and m gates, there are two obvious simulation algorithms. The first, which we could call the "Schrödinger" algorithm, stores the entire state vector in memory, using $\sim m2^n$ time and $\sim 2^n$ space. The second, which we could call the "Feynman" algorithm, calculates an amplitude as a sum of terms, using $\sim 4^m$ time and $\sim m+n$ space, as in the proof of $\mathsf{BQP} \subseteq \mathsf{P}^{\#\mathsf{P}}$ [18].

Now typically $m \gg n$, and the difference between m and n could matter enormously in practice. For example, in the planned Google setup, n will be roughly 40 or 50, while m will ideally be in the thousands. Thus, 2^n time is reasonable whereas 4^m time is not. So a question arises:

- *When $m \gg n$, is there a classical algorithm to simulate an n-qubit, m-gate quantum circuit using both $\mathrm{poly}(m,n)$ space and much less than $\exp(m)$ time – ideally, more like $\exp(n)$?*

We show an affirmative answer. In particular, inspired by the proof of Savitch's Theorem [52], we give a recursive, sum-of-products algorithm that uses $\mathrm{poly}(m,n)$ space and $m^{O(n)}$ time – or better yet, $d^{O(n)}$ time, where d is the circuit depth. We also show how to improve the running time further for quantum circuits subject to nearest-neighbor constraints, such as the superconducting systems currently under development. Finally, we show the existence of a "smooth tradeoff" between our algorithm and the 2^n-memory Schrödinger algorithm. Namely, starting with the Schrödinger algorithm, for every desired halving of the memory usage, one can multiply the running time by an additional factor of $\sim d$.

We hope our algorithm finds some applications in quantum simulation. In the meantime, though, the key point for this paper is that neither the Feynman algorithm, nor the Schrödinger algorithm, nor our new recursive algorithm come close to refuting QUATH. The Feynman algorithm fails to refute QUATH because it yields only a $1/\exp(m)$ advantage over random guessing, rather than a $1/2^n$ advantage.[4] The Schrödinger and recursive algorithms have much closer to the "correct" 2^n running time, but they also fail to refute QUATH because they don't calculate amplitudes as straightforward sums, so don't lead to polynomial-time guessing algorithms at all. Thus, in asking whether we can falsify QUATH, in some sense we're asking how far we can go in combining the advantages of all these algorithms. This might, in turn, connect to longstanding open problems about the optimality of Savitch's Theorem itself (e.g., L versus NL).

[4] Note that the Feynman algorithm can also be interpreted as a PP algorithm.

Interestingly, our analysis of quantum circuit simulation algorithms explains why this paper's hardness argument for quantum circuit sampling, based on QUATH, would *not* have worked for quantum supremacy proposals such as BosonSampling or IQP. It works only for the more general problem of quantum circuit sampling. The reason is that for the latter, unlike for BosonSampling or IQP, there exists a parameter $m \gg n$ (namely, the number of gates) that controls the advantage that a polynomial-time classical algorithm can achieve over random guessing, even while n controls the number of possible outputs. Our analysis also underscores the importance of taking $m \gg n$ in experiments meant to show quantum supremacy, and it provides some guidance to experimenters about the crucial question of what *circuit depth* they need for a convincing quantum supremacy demonstration.

Note that, the greater the required depth, the more protected against decoherence the qubits need to be. But the tradeoff is that the depth must be high enough that simulation algorithms that exploit limited entanglement, such as those based on tensor networks, are ruled out. Beyond that requirement, our $d^{O(n)}$ simulation algorithm gives some information about how much additional hardness one can purchase for a given increase in depth.

Strong Quantum Supremacy Theorems Must Be Non-Relativizing. Next, in Section 5, we switch our attention to a meta-question. Namely, what sorts of complexity-theoretic evidence we could possibly hope to offer for $\mathsf{SampBPP} \neq \mathsf{SampBQP}$: in other words, for quantum computers being able to solve approximate sampling problems that are hard classically? By Aaronson's sampling/searching equivalence theorem [8], any such evidence would *also* be evidence for $\mathsf{FBPP} \neq \mathsf{FBQP}$ (where FBPP and FBQP are the corresponding classes of relation problems), and vice versa.

Of course, an unconditional proof of these separations is out of the question right now, since it would imply $\mathsf{P} \neq \mathsf{PSPACE}$. Perhaps the next best thing would be to show that, if $\mathsf{SampBPP} = \mathsf{SampBQP}$, then the polynomial hierarchy collapses. This latter is *not* out of the question: as we said earlier, we already know, by a simple relativizing argument, that an equivalence between quantum and classical *exact* sampling implies the collapse $\mathsf{P}^{\#\mathsf{P}} = \mathsf{PH} = \mathsf{BPP}^{\mathsf{NP}}$. Furthermore, in their work on BosonSampling, Aaronson and Arkhipov [3] formulated a #P-hardness conjecture – namely, their so-called *Permanent of Gaussians Conjecture*, or PGC – that if true, would imply a generalization of that collapse to the physically relevant case of approximate sampling. More explicitly, Aaronson and Arkhipov showed that if the PGC holds, then

$$\mathsf{SampBPP} = \mathsf{SampBQP} \Longrightarrow \mathsf{P}^{\#\mathsf{P}} = \mathsf{BPP}^{\mathsf{NP}}. \tag{1}$$

They went on to propose a program for proving the PGC, by exploiting the random self-reducibility of the permanent. On the other hand, Aaronson and Arkhipov also explained in detail why new ideas would be needed to complete that program, and the challenge remains open.

Subsequently, Bremner, Montanaro, and Shepherd [24, 25] gave analogous #P-hardness conjectures that, if true, would *also* imply the implication (1), by going through the IQP model rather than through BosonSampling.

Meanwhile, nearly two decades ago, Fortnow and Rogers [30] exhibited an oracle relative to which $\mathsf{P} = \mathsf{BQP}$ and yet the polynomial hierarchy is infinite. In other words, they showed that any proof of the implication

$$\mathsf{P} = \mathsf{BQP} \Longrightarrow \mathsf{PH} \text{ collapses}$$

would have to be non-relativizing. Unfortunately, their construction was extremely specific to languages (i.e., total Boolean functions), and didn't even rule out the possibility that the

implication

$$\mathsf{PromiseBPP} = \mathsf{PromiseBQP} \implies \mathsf{PH collapses}$$

could be proven in a relativizing way. Thus, Aaronson and Arkhipov [3, see Section 10] raised the question of which quantum supremacy theorems hold relative to all oracles.

In Section 5, we fill in the final piece needed to resolve their question, by constructing an oracle A relative to which $\mathsf{SampBPP}^A = \mathsf{SampBQP}^A$ and yet PH^A is infinite. In other words, we show that *any strong supremacy theorem for quantum sampling, along the lines of what Aaronson and Arkhipov [3] and Bremner, Montanaro, and Shepherd [24, 25] were seeking, must use non-relativizing techniques.* In that respect, the situation with approximate sampling is extremely different from that with exact sampling.

Perhaps it's no surprise that one would need non-relativizing techniques to prove a strong quantum supremacy theorem. In fact, Aaronson and Arkhipov [3] were originally led to study BosonSampling precisely because of the connection between bosons and the permanent function, and the hope that one could therefore exploit the famous non-relativizing properties of the permanent to prove hardness. All the same, this is the first time we have explicit confirmation that non-relativizing techniques will be needed.

Maximal Quantum Supremacy for Black-Box Sampling and Relation Problems. In Section 6, we turn our attention to the black-box model, and specifically to the question: *what are the largest possible separations between randomized and quantum query complexities for any approximate sampling or relation problem?* Here we settle another open question. In 2015, Aaronson and Ambainis [9] studied Fourier Sampling, in which we're given access to a Boolean function $f : \{0,1\}^n \rightarrow \{0,1\}$, and the goal is to sample a string z with probability $\widehat{f}(z)^2$, where $\widehat{f}$ is the Boolean Fourier transform of f, normalized so that $\sum_z \widehat{f}(z)^2 = 1$. This problem is trivially solvable by a quantum algorithm with only 1 query to f. By contrast, Aaronson and Ambainis showed that there exists a constant $\varepsilon > 0$ such that any classical algorithm that solves Fourier Sampling, to accuracy ε in variation distance, requires $\Omega(2^n/n)$ queries to f. They conjectured that this lower bound was tight.

Here we refute that conjecture, by proving a $\Omega(2^n)$ lower bound on the randomized query complexity of Fourier Sampling, as long as ε is sufficiently small (say, $\frac{1}{40000}$). This implies that, for approximate sampling problems, the gap between quantum and randomized query complexities can be as large as imaginable: namely, *1 versus linear (!)*.[5] This sharply contrasts with the case of partial Boolean functions, for which Aaronson and Ambainis [9] showed that any N-bit problem solvable with k quantum queries is also solvable with $O\left(N^{1-1/2k}\right)$ randomized queries, and hence a constant versus linear separation is impossible. Thus, our result helps once again to underscore the advantage of sampling problems over decision problems for quantum supremacy experiments. Given the extremely close connection between Fourier Sampling and the IQP model [23], our result also provides some evidence that classically simulating an n-qubit IQP circuit, to within constant error in variation distance, is about as hard as can be: it might literally require $\Omega(2^n)$ time.

Aaronson and Ambainis [9] didn't directly address the natural relational version of Fourier Sampling, which Aaronson [1] had called Fourier Fishing in 2009. In Fourier Fishing,

[5] We have learned (personal communication) that recently, and independently of us, Ashley Montanaro has obtained a communication complexity result that implies this result as a corollary.

the goal is to output any string z such that $\widehat{f}(z)^2 \geq 1$, with nontrivial success probability. Unfortunately, the best lower bound on the randomized query complexity of Fourier Fishing that follows from [1] has the form $2^{n^{\Omega(1)}}$. As a further contribution, in Section 6 we give a lower bound of $\Omega(2^n/n)$ on the randomized query complexity of Fourier Fishing, which both simplifies and subsumes the $\Omega(2^n/n)$ lower bound for Fourier Sampling by Aaronson and Ambainis [9] (which, of course, we also improve to $\Omega(2^n)$ in this paper).

Quantum Supremacy Relative to Efficiently-Computable Oracles. In Section 7, we ask a new question: when proving quantum supremacy theorems, can we "interpolate" between the black-box setting of Sections 5 and 6, and the non-black-box setting of Sections 3 and 4? In particular, what happens if we consider quantum sampling algorithms that can access an oracle, *but* we impose a constraint that the oracle has to be "physically realistic"? One natural requirement here is that the oracle function f be computable in the class $\mathsf{P/poly}$:[6] in other words, that there are polynomial-size circuits for f, which we imagine that our sampling algorithms (both quantum and classical) can call as subroutines. If the sampling algorithms *also* had access to explicit descriptions of the circuits, then we'd be back in the computational setting, where we already know that there's no hope at present of proving quantum supremacy unconditionally. But what if our sampling algorithms know only that small circuits for f exist, without knowing what they are? Could quantum supremacy be proven unconditionally *then*?

We give a satisfying answer to this question. First, by adapting constructions due to Zhandry [61] and (independently) Servedio and Gortler [53], we show that if one-way functions exist, then there are oracles $A \in \mathsf{P/poly}$ such that $\mathsf{BPP}^A \neq \mathsf{BQP}^A$, and indeed even $\mathsf{BQP}^A \not\subset \mathsf{SZK}^A$. (Here and later, the one-way functions only need to be hard to invert classically, not quantumly.)

Note that, in the unrelativized world, there seems to be no hope at present of proving $\mathsf{BPP} \neq \mathsf{BQP}$ under any hypothesis nearly as weak as the existence of one-way functions. Instead one has to assume the one-wayness of extremely *specific* functions, for example those based on factoring or discrete log.

Second, and more relevant to near-term experiments, we show that if there exist one-way functions that take at least subexponential time to invert, then there are Boolean functions $f \in \mathsf{P/poly}$ such that approximate Fourier Sampling on those f's requires classical exponential time. In other words: within our "physically realistic oracle" model, there are feasible-looking quantum supremacy experiments, along the lines of the IQP proposal [23], such that a very standard and minimal cryptographic assumption is enough to prove the hardness of simulating those experiments classically.

Third, we show that the above two results are essentially optimal, by proving a converse result: that even in our $\mathsf{P/poly}$ oracle model, *some* computational assumption is still needed to prove quantum supremacy. The precise statement is this: if $\mathsf{SampBPP} = \mathsf{SampBQP}$ and $\mathsf{NP} \subseteq \mathsf{BPP}$, then $\mathsf{SampBPP}^A = \mathsf{SampBQP}^A$ for all $A \in \mathsf{P/poly}$. Or equivalently: if we want to separate quantum from classical approximate sampling relative to efficiently computable oracles, then we need to assume *something* about the unrelativized world: either $\mathsf{SampBPP} \neq \mathsf{SampBQP}$ (in which case we wouldn't even need an oracle), or else $\mathsf{NP} \not\subset \mathsf{BPP}$ (which is closely related to the assumption we *do* make, namely that one-way functions exist).

So to summarize, we've uncovered a "smooth tradeoff" between the model of computation and the hypothesis needed for quantum supremacy. Relative to *some* oracle (and even a

[6] More broadly, we could let f be computable in $\mathsf{BQP/poly}$, but this doesn't change the story too much.

random oracle), we can prove $\mathsf{SampBPP} \neq \mathsf{SampBQP}$ unconditionally. Relative to some efficiently computable oracle, we can prove $\mathsf{SampBPP} \neq \mathsf{SampBQP}$, but only under a weak computational assumption, like the existence of one-way functions. Finally, with no oracle, we can currently prove $\mathsf{SampBPP} \neq \mathsf{SampBQP}$ only under special assumptions, such as factoring being hard, or the permanents of Gaussian matrices being hard to approximate in $\mathsf{BPP}^{\mathsf{NP}}$, or our QUATH assumption. Perhaps eventually, we'll be able to prove $\mathsf{SampBPP} \neq \mathsf{SampBQP}$ under the sole assumption that PH is infinite, which would be a huge step forward – but at any rate we'll need *some* separation of classical complexity classes.[7]

One last remark: the idea of comparing complexity classes relative to $\mathsf{P/poly}$ oracles seems quite natural even apart from its applications to quantum supremacy. So in Appendix A, we take an initial stab at exploring the implications of that idea for other central questions in complexity theory. In particular, we prove the surprising result there that $\mathsf{P}^A = \mathsf{BPP}^A$ for all oracles $A \in \mathsf{P/poly}$, *if and only if* the derandomization hypothesis of Impagliazzo and Wigderson [36] holds (i.e., there exists a function in E with $2^{\Omega(n)}$ circuit complexity). In our view, this helps to clarify Impagliazzo and Wigderson's theorem itself, by showing precisely in what way their circuit lower bound hypothesis is stronger than the desired conclusion $\mathsf{P} = \mathsf{BPP}$. We also show that, if there are quantumly-secure one-way functions, then there exists an oracle $A \in \mathsf{P/poly}$ such that $\mathsf{SZK}^A \not\subset \mathsf{BQP}^A$.

1.4 Techniques

In our view, the central contributions of this work lie in the creation of new questions, models, and hardness assumptions (such as QUATH and quantum supremacy relative to $\mathsf{P/poly}$ oracles), as well as in basic observations that somehow weren't made before (such as the sum-products algorithm for simulating quantum circuits) – all of it motivated by the goal of using complexity theory to inform ongoing efforts in experimental physics to test the Extended Church-Turing Thesis. While some of our proofs are quite involved, by and large the proof techniques are ones that will be familiar to complexity theorists. Even so, it seems appropriate to say a few words about techniques here.

To prove, in Section 3, that "if QUATH is true, then HOG is hard, textquotedblright we give a fairly straightforward reduction: namely, we assume the existence of a polynomial-time classical algorithm to find high-probability outputs of a given quantum circuit C. We then use that algorithm (together with a random self-reduction trick) to guess the magnitude of a particular transition amplitude, such as $\langle 0^n|C|0^n\rangle$, with probability slightly better than chance, which is enough to refute QUATH.

One technical step is to show that, with $\Omega(1)$ probability, the distribution over n-bit strings sampled by a random quantum circuit C is far from the uniform distribution. But not only can this be done, we show that it can be done by examining only the very last gate of C, and ignoring all other gates! A challenge that we leave open is to improve this, to show that the distribution sampled by C is far from uniform, not merely with $\Omega(1)$ probability, but with $1 - 1/\exp(n)$ probability. In Appendix E, we present numerical evidence for this conjecture, and indeed for a stronger conjecture, that the probabilities appearing in the output distribution of a random quantum circuit behave like independent, exponentially-distributed random variables. (We note that Brandao, Harrow and Horodecki [21] recently proved a closely-related result, which unfortunately is not quite strong enough for our purposes.)

In Section 4, to give our polynomial-space, $d^{O(n)}$-time classical algorithm for simulating an n-qubit, depth-d quantum circuit C, we use a simple recursive strategy, reminiscent of

[7] Unless, of course, someone were to separate P from PSPACE unconditionally!

Savitch's Theorem. Namely, we slice the circuit into two layers, C_1 and C_2, of depth $d/2$ each, and then express a transition amplitude $\langle x|C|z\rangle$ of interest to us as

$$\langle x|C|z\rangle = \sum_{y\in\{0,1\}^n} \langle x|C_1|y\rangle\langle y|C_2|z\rangle.$$

We then compute each $\langle x|C_1|y\rangle$ and $\langle y|C_2|z\rangle$ by recursively slicing C_1 and C_2 into layers of depth $d/4$ each, and so on. What takes more work is to obtain a further improvement if C has only nearest-neighbor interactions on a grid graph – for that, we use a more sophisticated divide-and-conquer approach – and also to interpolate our recursive algorithm with the 2^n-space Schrödinger simulation, in order to make the best possible use of whatever memory is available.

Our construction, in Section 5, of an oracle relative to which SampBPP = SampBQP and yet PH is infinite involves significant technical difficulty. As a first step, we can use a PSPACE oracle to collapse SampBPP with SampBQP, and then use one of many known oracles (or, by the recent breakthrough of Rossman, Servedio, and Tan [49], even a *random* oracle) to make PH infinite. The problem is that, if we do this in any naïve way, then the oracle that makes PH infinite will also re-separate SampBPP and SampBQP, for example because of the approximate Fourier Sampling problem. Thus, we need to *hide* the oracle that makes PH infinite, in such a way that a PH algorithm can still find the oracle (and hence, PH is still infinite), but a SampBQP algorithm can't find it with any non-negligible probability – crucially, not even if the SampBQP algorithm's input x provides a clue about the oracle's location. Once one realizes that these are the challenges, one then has about seven pages of work to ensure that SampBPP and SampBQP remain equal, relative to the oracle that one has constructed. Incidentally, we know that this equivalence can't possibly hold for *exact* sampling, so *something* must force small errors to arise when the SampBPP algorithm simulates the SampBQP one. That something is basically the tiny probability that the quantum algorithm will succeed at finding the hidden oracle, which however can be upper-bounded using quantum-mechanical linearity.

In Section 6, to prove a $\Omega(2^n)$ lower bound on the classical query complexity of approximate Fourier Sampling, we use the same basic strategy that Aaronson and Ambainis [9] used to prove a $\Omega(2^n/n)$ lower bound, but with a much more careful analysis. Specifically, we observe that any Fourier Sampling algorithm would also yield an algorithm whose probability of accepting, while always small, is extremely sensitive to some specific Fourier coefficient, say $\widehat{f}(0\cdots 0)$. We then lower-bound the randomized query complexity of accepting with the required sensitivity to $\widehat{f}(0\cdots 0)$, taking advantage of the fact that $\widehat{f}(0\cdots 0)$ is simply proportional to $\sum_x f(x)$, so that all x's can be treated symmetrically. Interestingly, we also give a different, much simpler argument that yields a $\Omega(2^n/n)$ lower bound on the randomized query complexity of Fourier Fishing, which then immediately implies a $\Omega(2^n/n)$ lower bound for Fourier Sampling as well. However, if we want to improve the bound to $\Omega(2^n)$, then the original argument that Aaronson and Ambainis [9] used to prove $\Omega(2^n/n)$ seems to be needed.

In Section 7, to prove that one-way functions imply the existence of an oracle $A \in \mathsf{P/poly}$ such that $\mathsf{P}^A \neq \mathsf{BQP}^A$, we adapt a construction that was independently proposed by Zhandry [61] and by Servedio and Gortler [53]. In this construction, we first use known reductions [34, 32] to convert a one-way function into a classically-secure pseudorandom permutation, say σ. We then define a new function by $g_r(x) := \sigma(x \bmod r)$, where x is interpreted as an integer written in binary, and r is a hidden period. Finally, we argue that either Shor's

algorithm [55] leads to a quantum advantage over classical algorithms in finding the period of g_r, or else g_r was not pseudorandom, contrary to assumption. To show that subexponentially-secure one-way functions imply the existence of an oracle $A \in \mathsf{P/poly}$ relative to which Fourier Sampling is classically hard, we use similar reasoning. The main difference is that now, to construct a distinguisher against a pseudorandom function f, we need classical exponential time just to *verify* the outputs of a claimed polynomial-time classical algorithm for Fourier Sampling f – and that's why we need to assume $2^{n^{\Omega(1)}}$ security.

Finally, to prove that $\mathsf{SampBPP} = \mathsf{SampBQP}$ and $\mathsf{NP} \subseteq \mathsf{BPP}$ imply $\mathsf{SampBPP}^A = \mathsf{SampBQP}^A$ for all $A \in \mathsf{P/poly}$, we design a step-by-step classical simulation of a quantum algorithm, call it Q, that queries an oracle $A \in \mathsf{P/poly}$. We use the assumption $\mathsf{SampBPP} = \mathsf{SampBQP}$ to sample from the probability distribution over queries to A that Q makes at any given time step. Then we use the assumption $\mathsf{NP} \subseteq \mathsf{BPP}$ to guess a function $f \in \mathsf{P/poly}$ that's consistent with $n^{O(1)}$ sampled classical queries to A. Because of the limited number of functions in $\mathsf{P/poly}$, standard sample complexity bounds for PAC-learning imply that any such f that we guess will probably agree with the "true" oracle A on most inputs. Quantum-mechanical linearity then implies that the rare disagreements between f and A will have at most a small effect on the future behavior of Q.

2 Preliminaries

For a positive integer n, we use $[n]$ to denote the integers from 1 to n. Logarithms are base 2.

2.1 Quantum Circuits

We now introduce some notations for quantum circuits, which will be used throughout this paper.

In a quantum circuit, without loss of generality, we assume all gates are unitary and acting on exactly two qubits each[8].

Given a quantum circuit C, slightly abusing notation, we also use C to denote the unitary operator induced by C. Suppose there are n qubits and m gates in C; then we index the qubits from 1 to n. We also index gates from 1 to m in chronological order for convenience.

For each subset $S \subseteq [n]$ of the qubits, let $\mathcal{H}_S$ be the Hilbert space corresponding to the qubits in S, and I_S be the identity operator on $\mathcal{H}_S$. Then the unitary operator U_i for the i-th gate can be written as $U_i := O_i \otimes I_{[n]\setminus\{a_i,b_i\}}$, in which O_i is a unitary operator on $\mathcal{H}_{\{a_i,b_i\}}$ (the Hilbert space spanned by the qubits a_i and b_i), and $I_{[n]\setminus\{a_i,b_i\}}$ is the identity operator on the other qubits.

We say that a quantum circuit has depth d, if its gates can be partitioned into d layers (in chronological order), such that the gates in each layer act on disjoint pairs of qubits. Suppose the i-th layer consists of the gates in $[L_i, R_i]$. We define $C_{[r\leftarrow l]} = U_{R_r} \cdot U_{R_r-1} \ldots U_{L_l+1} \cdot U_{L_l}$, that is, the sub-circuit between the l-th layer and the r-th layer.

Base Graphs and Grids

In Sections 3 and 4, we will sometimes assume *locality* of a given quantum circuit. To formalize this notion, we define the *base graph* of a quantum circuit.

[8] Except for oracle gates, which may act on any number of qubits.

▶ **Definition 1.** Given a quantum circuit C on n qubits, its base graph $G_C = (V, E)$ is an undirected graph defined by $V = [n]$, and

$E = \{(a, b) \mid$ there is a quantum gate that acts on qubits a and $b.\}$.

We will consider a specific kind of base graph, the grids.

▶ **Definition 2.** The grid G of size $H \times W$ is a graph with vertices $V = \{(x, y) \mid x \in [H], y \in [W]\}$ and edges $E = \{(a, b) \mid |a - b|_1 = 1, a \in V, b \in V\}$, and we say that grid G has H rows and W columns.

2.2 Complexity Classes for Sampling Problems

Definitions for SampBPP and SampBQP

We adopt the following definition for sampling problems from [8].

▶ **Definition 3** (Sampling Problems, SampBPP, and SampBQP)**.** A *sampling problem* S is a collection of probability distributions $(\mathcal{D}_x)_{x\in\{0,1\}^*}$, one for each input string $x \in \{0,1\}^n$, where $\mathcal{D}_x$ is a distribution over $\{0,1\}^{p(n)}$, for some fixed polynomial p. Then SampBPP is the class of sampling problems $S = (\mathcal{D}_x)_{x\in\{0,1\}^*}$ for which there exists a probabilistic polynomial-time algorithm B that, given $\left\langle x, 0^{1/\varepsilon} \right\rangle$ as input, samples from a probability distribution $\mathcal{C}_x$ such that $\|\mathcal{C}_x - \mathcal{D}_x\| \leq \varepsilon$. SampBQP is defined the same way, except that B is a quantum algorithm rather than a classical one.

Oracle versions of these classes can also be defined in the natural way.

A Canonical Form of SampBQP Oracle Algorithms

To ease our discussion about $\mathsf{SampBQP}^{\mathcal{O}}$, we describe a canonical form of SampBQP oracle algorithms. Any other reasonable definitions of SampBQP oracle algorithms (like with quantum oracle Turing machines) can be transformed into this form easily.

Without loss of generality, we can assume a SampBQP oracle algorithm M with oracle access to $\mathcal{O}_1, oracle_2, dotsc, oracle_k$ (k is a universal constant) acts in three stages, as follows.

1. Given an input $\langle x, 0^{1/\varepsilon}\rangle$, M first uses a classical routine (which does not use the oracles) to output a quantum circuit C with $p(n, 1/\varepsilon)$ qubits and $p(n, 1/\varepsilon)$ gates in polynomial time, where p is a fixed polynomial. Note that C can use the $\mathcal{O}_1, oracle_2, dotsc, oracle_k$ gates in addition to a universal set of quantum gates.
2. Then M runs the outputted quantum circuit with the initial state $|0\rangle^{\otimes p(n,1/\varepsilon)}$, and measures all the qubits to get an outcome z in $\{0,1\}^{p(n,1/\varepsilon)}$.
3. Finally, M uses another classical routine A^{output} (which does not use the oracles) on the input z, to output its final sample $A^{\mathsf{output}}(z) \in \{0,1\}^*$.

Clearly, M solves different sampling problems (or does not solve any sampling problem at all) given different oracles $\mathcal{O}_1, oracle_2, dotsc, oracle_k$. Therefore, we use $M^{\mathcal{O}_1, oracle_2, dotsc, oracle_k}$ to indicate the particular algorithm when the oracles are $\mathcal{O}_1, oracle_2, dotsc, oracle_k$.

2.3 Distinguishing Two Pure Quantum States

We also need a standard result for distinguishing two pure quantum states.

▶ **Theorem 4** (Helstrom's decoder for two pure states). *The maximum success probability for distinguishing two pure quantum states $|\varphi_0\rangle$ and $|\varphi_1\rangle$ given with prior probabilities π_0 and π_1, is given by*

$$p_{succ} = \frac{1+\sqrt{1-4\pi_0\pi_1 F}}{2},$$

where $F := |\langle\varphi_0|\varphi_1\rangle|^2$ is the fidelity between the two states.

We'll also need that for two similar quantum states, the distributions induced by measuring them are close.

▶ **Corollary 5.** *Let $|\varphi_0\rangle$ and $|\varphi_1\rangle$ be two pure quantum state such that $||\varphi_0\rangle - |\varphi_1\rangle| \leq \varepsilon$. For a quantum state φ, define $\mathcal{D}(\varphi)$ be the distribution on $\{0,1\}^*$ induced by some quantum sampling procedure, we have*

$$\|\mathcal{D}(\varphi_0) - \mathcal{D}(\varphi_1)\| \leq \sqrt{2\varepsilon}.$$

Proof. Fix prior probabilities $\pi_0 = \pi_1 = \frac{1}{2}$.

Note that we have a distinguisher of $|\varphi_0\rangle$ and $|\varphi_1\rangle$ with success probability

$$\frac{1+\|\mathcal{D}(\varphi_0) - \mathcal{D}(\varphi_1)\|}{2}$$

by invoking that quantum sampling procedure.

By the assumption, $|\langle\varphi_0||\varphi_1\rangle| = |\langle\varphi_0|\cdot(|\varphi_0\rangle+(|\varphi_1\rangle-|\varphi_0\rangle))| \geq 1-\varepsilon$, hence $F = |\langle\varphi_0|\varphi_1\rangle|^2 \geq (1-\varepsilon)^2$. So we have

$$\frac{1+\|\mathcal{D}(\varphi_0) - \mathcal{D}(\varphi_1)\|}{2} \leq \frac{1+\sqrt{1-(1-\varepsilon)^2}}{2}.$$

This implies $\|\mathcal{D}(\varphi_0) - \mathcal{D}(\varphi_1)\|_1 \leq \sqrt{1-(1-\varepsilon)^2} = \sqrt{2\varepsilon-\varepsilon^2} \leq \sqrt{2\varepsilon}$. ◀

2.4 A Multiplicative Chernoff Bound

▶ **Lemma 6.** *Suppose $X_1, X_2, dotsc, X_n$ are independent random variables taking values in $[0,1]$. Let X denote their sum and let $\mu = \mathbb{E}[X]$. Then for any $\delta > 1$, we have*

$$\Pr[X \geq (1+\delta)\mu] \leq e^{-\frac{\delta\mu}{3}}.$$

▶ **Corollary 7.** *For any $0 < \tau$, suppose $X_1, X_2, dotsc, X_n$ are independent random variables taking values in $[0, tau]$. Let X denote their sum and let $\mu = \mathbb{E}[X]$. Then for any $\delta > 1$, we have*

$$\Pr[X \geq (1+\delta)\mu] \leq e^{-\frac{\delta\mu}{3\tau}}.$$

Proof. Replace each X_i by X_i/τ and apply the previous lemma. ◀

3 The Hardness of Quantum Circuit Sampling

We now discuss our random quantum circuit proposal for demonstrating quantum supremacy.

3.1 Preliminaries

We first introduce some notations. We use $\mathbb{U}(N)$ to denote the group of $N \times N$ unitary matrices, μ^N_{Haar} for the Haar measure on $\mathbb{U}(N)$, and μ^N_{rand} for the Haar measure on N-dimensional pure states.

For a pure state $|u\rangle$ on n qubits, we define $\mathsf{probList}(|u\rangle)$ to be the list consisting of 2^n numbers, $|\langle u|x\rangle|^2$ for each $x \in \{0,1\}^n$.

Given N real numbers $a_1, a_2, dotsc, a_N$, we use $\mathsf{uphalf}(a_1, a_2, dotsc, a_N)$ to denote the sum of the largest $N/2$ numbers among them, and we let

$$\mathsf{adv}(|u\rangle) = \mathsf{uphalf}(\mathsf{probList}(|u\rangle)).$$

Finally, we say that an output $z \in \{0,1\}^n$ is *heavy* for a quantum circuit C, if it is greater than the median of $\mathsf{probList}(C|0^n\rangle)$.

3.2 Random quantum circuit on grids

Recall that we assume a quantum circuit consists of only 2-qubit gates. Our random quantum circuit on grids of n qubits and m gates (assuming $m \geq n$) is generated as follows (though the basic structure of our hardness argument will not be very sensitive to details, and would also work for many other circuit ensembles):

- All the qubits are arranged as a $\sqrt{n} \times \sqrt{n}$ grid (see Definition 2), and a gate can only act on two adjacent qubits.
- For each $t \in [m]$ with $t \leq n$, we pick the t-th qubit and a random neighbor of it.[9]
- For each $t \in [m]$ with $t > n$, we pick a uniform random pair of adjacent qubits in the grid $\sqrt{n} \times \sqrt{n}$.
- Then, in either case, we set the t-th gate to be a unitary drawn from μ^4_{Haar} acting on these two qubits.

Slightly abusing notation, we use $\mu^{n,m}_{\mathsf{grid}}$ to denote both the above distribution on quantum circuits and the distribution on $\mathbb{U}(2^n)$ induced by it.

Conditional distribution ν_{grid}

For convenience, for a quantum circuit C, we abbreviate $\mathsf{adv}(C|0^n\rangle)$ as $\mathsf{adv}(C)$. Consider a simple quantum algorithm which measures $C|0^n\rangle$ in the computational basis to get an output z. Then by definition, $\mathsf{adv}(C)$ is simply the probability that z is heavy for C.

We want that, when a quantum circuit C is drawn, $\mathsf{adv}(C)$ is *large* (that is, bounded above $1/2$), and therefore the simple quantum algorithm has a substantial advantage on generating a heavy output, compared with the trivial algorithm of guessing a random string.

For convenience, we also consider the following conditional distribution $\nu^{n,m}_{\mathsf{grid}}$: it keeps drawing a circuit $C \leftarrow \mu^{n,m}_{\mathsf{grid}}$ until the sample circuit C satisfies $\mathsf{adv}(C) \geq 0.7$.

Lower bound on $\mathsf{adv}(C)$

We need to show that a circuit C drawn from $\nu^{n,m}_{\mathsf{grid}}$ has a large probability of having $\mathsf{adv}(C) \geq 0.7$. In order to show that, we give a cute and simple lemma, which states that the *expectation* of $\mathsf{adv}(C)$ is *large*. Surprisingly, its proof only makes use of the randomness introduced by the *very last gate*!

[9] The purpose here is to make sure that there is a gate on every qubit.

▶ **Lemma 8.** *For $n \geq 2$ and $m \geq n$*

$$\mathop{\mathbb{E}}_{C \leftarrow \mu_{\mathsf{grid}}^{n,m}} [\mathsf{adv}(C)] \geq \frac{5}{8}.$$

In fact, we conjecture that $\mathsf{adv}(C)$ is large with an *overwhelming* probability.

▶ **Conjecture 1.** *For $n \geq 2$ and $m \geq n^2$, and for all constants $\varepsilon > 0$,*

$$\Pr_{C \leftarrow \mu_{\mathsf{grid}}^{n,m}} \left[\mathsf{adv}(C) < \frac{1 + \ln 2}{2} - \varepsilon \right] < \exp\{-\Omega(n)\}.$$

We give some numerical simulation evidence for Conjecture 1 in Appendix E.

▶ Remark 9. Assuming Conjecture 1, in practice, one can sample from ν_{grid} by simply sampling from μ_{grid}, the uniform distribution over circuits—doing so only introduces an error probability of $\exp\{-\Omega(n)\}$.

3.3 The HOG Problem

Now we formally define the task in our quantum algorithm proposal.

▶ **Problem 2** (HOG, or Heavy Output Generation)**.** *Given a random quantum circuit C from $\nu_{\mathsf{grid}}^{n,m}$ for $m \geq n^2$, generate k binary strings $z_1, z_2, dotsc, z_k$ in $\{0,1\}^n$ such that at least a $2/3$ fraction of z_i's are heavy for C.*

The following proposition states that there is a simple quantum algorithm which solves the above problem with overwhelming probability.

▶ **Proposition 10.** *There is a quantum algorithm that succeeds at HOG with probability $1 - \exp\{-\Omega(k)\}$.*

Proof. The algorithm just simulates the circuit C with initial state $|0^n\rangle$, then measures in the computational basis k times independently to output k binary strings.

From the definition of ν_{grid}, we have $\mathsf{adv}(C) \geq 0.7 > 2/3$. So by a Chernoff bound, with probability $1 - \exp\{\Omega(k)\}$, at least a $2/3$ fraction of z_i's are heavy for C, in which case the algorithm solves HOG. ◀

3.4 Classical Hardness Assuming QUATH

We now state our classical hardness assumption.

▶ **Assumption 2** (QUATH, or the **Qua**ntum **Th**reshold assumption)**.** *There is no polynomial-time classical algorithm that takes as input a random quantum circuit $C \leftarrow \nu_{\mathsf{grid}}^{n,m}$ for $m \geq n^2$ and decides whether 0^n is heavy for C with success probability $1/2 + \Omega(2^{-n})$.*

▶ Remark 11. Note that $1/2$ is the success probability obtained by always outputting either 0 or 1. Therefore, the above assumption means that no efficient algorithm can beat the trivial algorithm even by $\Omega(2^{-n})$.

Next, we show that QUATH implies that no efficient classical algorithm can solve HOG.

▶ **Theorem 12.** *Assuming QUATH, no polynomial-time classical algorithm can solve HOG with probability at least* 0.99*.*

Proof. Suppose by contradiction that there is such a classical polynomial-time algorithm A. Using A, we will construct an algorithm to violate QUATH.

The algorithm is quite simple. Given a quantum circuit $C \leftarrow \nu_{\mathsf{grid}}^{n,m}$, we first draw a uniform random string $z \in \{0,1\}^n$. Then for each i such that $z_i = 1$, we apply a NOT gate on the i-th qubit. Note that this gate can be "absorbed" into the last gate acting on the i-th qubit in C. Hence, we still get a circuit C' with m gates. Moreover, it is easy to see that C' is distributed exactly the same as C even if conditioning on a particular z, and we have $\langle 0^n|C|0^n\rangle = \langle 0^n|C'|z\rangle$, which means that 0^n is heavy for C if and only if z is heavy for C'.

Next our algorithm runs A on circuit C' to get k outputs $z_1, dotsc, z_k$, and picks an output $z_{i^\star}$ among these k outputs uniformly at random. If $z_{i^\star} = z$, then the algorithm outputs 1; otherwise it outputs a uniform random bit.

Since A solves HOG with probability 0.99, we have that each z_k is heavy for C' with probability at least $0.99 \cdot 2/3$.

Now, since z is a uniform random string, the probability that our algorithm decides correctly whether z is heavy for C' is

$$\Pr[z = z_{i^\star}] \cdot 0.99 \cdot \frac{2}{3} + \Pr[z \neq z_{i^\star}] \cdot 1/2 = 2^{-n} \cdot 0.99 \cdot \frac{2}{3} + (1 - 2^{-n}) \cdot 1/2$$
$$= \frac{1}{2} + \Omega(2^{-n}).$$

But this contradicts QUATH, so we are done. ◀

3.5 Proof for Lemma 8

We first need a simple lemma which helps us to lower bound $\mathsf{adv}(|u\rangle)$.

For a pure quantum state $|u\rangle$, define

$$\mathsf{dev}(|u\rangle) = \sum_{w \in \{0,1\}^n} \left| |\langle u|w\rangle|^2 - 2^{-n} \right|.$$

In other words, $\mathsf{dev}(|u\rangle)$ measures the *non-uniformity* of the distribution obtained by measuring $|u\rangle$ in the computational basis.

The next lemma shows that, when $\mathsf{dev}(|u\rangle)$ is large, so is $\mathsf{adv}(|u\rangle)$. Therefore, in order to establish Lemma 8, it suffices to lower-bound $\mathsf{dev}(|u\rangle)$.

▶ **Lemma 13.** *For a pure quantum state $|u\rangle$, we have*

$$\mathsf{adv}(|u\rangle) \geq \frac{1}{2} + \frac{\mathsf{dev}(u)}{4}.$$

We will also need the following technical lemma.

▶ **Lemma 14.** *Let $|u\rangle \leftarrow \mu_{\mathsf{rand}}^2$. Then*

$$\mathop{\mathbb{E}}_{|u\rangle \leftarrow \mu_{\mathsf{rand}}^2} \left[\left| |\langle u|0\rangle|^2 - |\langle u|1\rangle|^2 \right| \right] = 0.5.$$

The proofs of Lemma 13 and Lemma 14 are based on simple but tedious calculations, so we defer them to Appendix B.

Now we are ready to prove Lemma 8.

Proof of Lemma 8. Surprisingly, our proof only uses the randomness introduced by the very last gate. That is, the claim holds even if there is an adversary who fixes all the gates except for the last one.

We use I_n to denote the n-qubit identity operator.
Let $C \leftarrow \mu_{\mathsf{grid}}^{n,m}$. From Lemma 13, it suffices to show that

$$\mathop{\mathbb{E}}_{C\leftarrow\mu_{\mathsf{grid}}^{n,m}}[\mathsf{dev}(C|0^n\rangle)] \geq \frac{1}{2}.$$

Suppose the last gate $U \leftarrow \mu_{\mathsf{Haar}}^4$ acts on qubits a and b. Let the unitary corresponding to the circuit before applying the last gate be V, and $|v\rangle = V|0^n\rangle$. Now, suppose we apply another unitary U_a drawn from μ_{Haar}^2 on the qubit a. It is not hard to see that U and $(U_a \otimes I_1) \cdot U$ are identically distributed. So it suffices to show that

$$\mathop{\mathbb{E}}_{U\leftarrow\mu_{\mathsf{Haar}}^4, U_a\leftarrow\mu_{\mathsf{Haar}}^2}\Big[\mathsf{adv}\Big((U_a \otimes I_{n-1})(U \otimes I_{n-2})|v\rangle\Big)\Big] \geq 0.6.$$

We are going to show that the above holds even for a fixed U. That is, fix a $U \in \mathbb{U}(4)$ and let $|u\rangle = U \otimes I_{n-2})|v\rangle$. Then we will prove that

$$\mathop{\mathbb{E}}_{U_a\leftarrow\mu_{\mathsf{Haar}}^2}\Big[\mathsf{dev}\Big((U_a \otimes I_{n-1})|v\rangle\Big)\Big] \geq \frac{1}{2}.$$

Without loss of generality, we can assume that a is the last qubit. Then we write

$$|u\rangle = \sum_{w\in\{0,1\}^n} a_w|w\rangle,$$

and

$$|z\rangle = (U_a \otimes I_{n-1})|u\rangle.$$

Now we partition the 2^n basis states into 2^{n-1} buckets, one for each string in $\{0,1\}^{n-1}$. That is, for each $p \in \{0,1\}^{n-1}$, there is a bucket that consists of basis states $\{|p0\rangle, spzp1\}$. Note that since U_a acts on the last qubit, only amplitudes of basis states in the same bucket can affect each other.

For a given $p \in \{0,1\}^{n-1}$, if both a_{p0} and a_{p1} are zero, we simply ignore this bucket. Otherwise, we can define a quantum state

$$|t_p\rangle = \frac{a_{p0}|0\rangle + a_{p1}|1\rangle}{\sqrt{|a_{p0}|^2 + |a_{p1}|^2}},$$

and

$$|z_p\rangle = U_a|t_p\rangle.$$

Clearly, we have $\langle z|p0\rangle = \sqrt{|a_{p0}|^2 + |a_{p1}|^2} \cdot \langle z_p|0\rangle$ and $\langle z|p1\rangle = \sqrt{|a_{p0}|^2 + |a_{p1}|^2} \cdot \langle z_p|1\rangle$. Plugging in, we have

$$\begin{aligned}
&\mathop{\mathbb{E}}_{U_a\leftarrow\mu_{\mathsf{Haar}}^2}\Big[\Big||\langle z|p0\rangle|^2 - 2^{-n}\Big| + \Big||\langle z|p1\rangle|^2 - 2^{-n}\Big|\Big] \\
\geq &\mathop{\mathbb{E}}_{U_a\leftarrow\mu_{\mathsf{Haar}}^2}\Big[\Big||\langle z|p0\rangle|^2 - |\langle z|p1\rangle|^2\Big|\Big] && \text{(triangle inequality)} \\
= &\big(|a_{p0}|^2 + |a_{p1}|^2\big) \cdot \mathop{\mathbb{E}}_{U_a\leftarrow\mu_{\mathsf{Haar}}^2}\Big[\Big||\langle z_p|0\rangle|^2 - |\langle z_p|1\rangle|^2\Big|\Big].
\end{aligned}$$

Now, since $|t_p\rangle$ is a pure state, and U_a is drawn from μ^2_{Haar}, we see that $|z_p\rangle$ is distributed as a Haar-random pure state. So from Lemma 14, we have

$$\mathop{\mathbb{E}}_{U_a \leftarrow \mu^2_{\mathsf{Haar}}} \left[\left| |\langle z_p|0\rangle|^2 - |\langle z_p|1\rangle|^2 \right| \right] = 0.5.$$

Therefore,

$$\mathop{\mathbb{E}}_{U_a \leftarrow \mu^2_{\mathsf{Haar}}} \left[\left| |\langle z|p0\rangle|^2 - 2^{-n} \right| + \left| |\langle z|p1\rangle|^2 - 2^{-n} \right| \right] \geq \frac{1}{2} \cdot \left(|a_{p0}|^2 + |a_{p1}|^2 \right).$$

Summing up for each $p \in \{0,1\}^{n-1}$, we have

$$\mathop{\mathbb{E}}_{U_a \leftarrow \mu^2_{\mathsf{Haar}}} [\mathsf{dev}(|z\rangle)] \geq \frac{1}{2},$$

which completes the proof. ◀

4 New Algorithms to Simulate Quantum Circuits

In this section, we present two algorithms for simulating a quantum circuit with n qubits and m gates: one algorithm for arbitrary circuits, and another for circuits that act locally on grids. What's new about these algorithms is that they use both polynomial space and close to $\exp(n)$ time (but despite that, they don't violate the QUATH assumption from Section 3, for the reason pointed out in Section 1.3). Previously, it was known how to simulate a quantum circuit in polynomial space and $\exp(m)$ time (as in the proof of $\mathsf{BQP} \subseteq \mathsf{P}^{\#\mathsf{P}}$), or in exponential space and $\exp(n)$ time.

In addition, we provide a time-space trade-off scheme, which enables even faster simulation at the cost of more space usage. See Section 2.1 for the quantum circuit notations that are used throughout this section.

4.1 Polynomial-Space Simulation Algorithms for General Quantum Circuits

We first present a simple recursive algorithm for general circuits.

▶ **Theorem 15.** *Given a quantum circuit C on n qubits with depth d, and two computational basis states $|x\rangle, spzy$, we can compute $\langle y|C|x\rangle$ in $O(n \cdot (2d)^{n+1})$ time and $O(n \log d)$ space.*

Proof. In the base case $d = 1$, the answer can be trivially computed in $O(n)$ time.

When $d > 1$, we have

$$\begin{aligned}
\langle y|C|x\rangle &= \langle y|C_{[d\leftarrow d/2+1]} \cdot C_{[d/2\leftarrow 1]}|x\rangle \\
&= \langle y|C_{[d\leftarrow d/2+1]} \left(\sum_{z\in\{0,1\}^n} |z\rangle\langle z| \right) C_{[d/2\leftarrow 1]}|x\rangle \\
&= \sum_{z\in\{0,1\}^n} \langle y|C_{[d\leftarrow d/2+1]}|z\rangle \cdot \langle z|C_{[d/2\leftarrow 1]}|x\rangle. \qquad (2)
\end{aligned}$$

Then, for each z, we calculate $\langle y|C_{[d\leftarrow d/2+1]}|z\rangle \cdot \langle z|C_{[d/2\leftarrow 1]}|x\rangle$ by recursively calling the algorithm on the two sub-circuits $C_{[d\leftarrow d/2+1]}$ and $C_{[d/2\leftarrow 1]}$ respectively; and sum them up to calculate (2).

It is easy to see the above algorithm is correct, and its running time can be analyzed as follows: let $F(d)$ be its running time on a circuit of d layers; then we have $F(1) = O(n)$, and by the above discussion

$$F(d) \leq 2^{n+1} \cdot F(\lceil d/2 \rceil) = O(n \cdot 2^{(n+1)\lceil \log d \rceil}) = O(n \cdot (2^{\lceil \log d \rceil})^{n+1}) \leq O(n \cdot (2d)^{n+1}),$$

which proves our running time bound.

Finally, we can see in each recursion level, we need $O(n)$ space to save the indices of $|x\rangle$ and $|y\rangle$, and $O(1)$ space to store an intermediate answer. Since there are at most $O(\log d)$ recursion levels, the total space is bounded by $O(n \log d)$. ◀

4.2 Faster Polynomial Space Simulation Algorithms for Grid Quantum Circuits

When a quantum circuit is spatially local, i.e., its base graph can be embedded on a grid, we can further speed up the simulation with a more sophisticated algorithm.

We first introduce a simple lemma which shows that we can find a small balanced cut in a two-dimensional grid.

▶ **Lemma 16.** *Given a grid $G = (V, E)$ of size $H \times W$ such that $|V| \geq 2$, we can find a subset $S \subset E$ such that*

- *$|S| \leq O(\sqrt{|V|})$, and*
- *after S is removed, G becomes a union of two disconnected grids with size smaller than $\frac{2}{3}|V|$.*

Proof. We can assume $H \geq W$ without loss of generality and simply set S to be the set of all the edges between the $\lfloor H/2 \rfloor$-th row and the $\lfloor H/2 \rfloor + 1$-th row; then both claims are easy to verify. ◀

We now present a faster algorithm for simulating quantum circuits on grids.

▶ **Theorem 17.** *Given a quantum circuit C on n qubits with depth d, and two computational basis states $|x\rangle, spzy$, assuming that G_C can be embedded into a two-dimensional grid with size n (with the embedding explicitly specified), we can compute $\langle y|C|x\rangle$ in $2^{O(d\sqrt{n})}$ time and $O(d \cdot n \log n)$ space.*

Proof. For ease of presentation, we slightly generalize the definition of quantum circuits: now each gate can be of the form $O_i \otimes I_{[n]\setminus\{a_i,b_i\}}$ (a 2-qubit gate) or $O_i \otimes I_{[n]\setminus\{a_i\}}$ (a 1-qubit gate) or simply $I_{[n]}$ (a 0-qubit gate, which is introduced just for convenience).

The algorithm works by trying to break the current large instance into many small instances which we then solve recursively. But unlike the algorithm in Theorem 15, which reduces an instance to many sub-instances with fewer *gates*, our algorithm here reduces an instance to many sub-instances with fewer *qubits*.

The base case, $n = 1$ qubit. In this case, all the gates are either 1-qubit or 0-qubit; hence the answer can be calculated straightforwardly in $O(m)$ time and constant space.

Cutting the grid by a small set. When $n \geq 2$, by Lemma 16, we can find a subset S of edges with $|S| \leq O(\sqrt{n})$. After S is removed, the grid becomes a union of two disconnected grids A and B (we use A, B to denote both the grids and the sets of the vertices in the grid for simplicity) with size smaller than $\frac{2}{3}n$.

Let

$$\{R = i \mid U_i \text{ is of the form } O_i \otimes I_{[n]\setminus\{a_i,b_i\}} \text{ and } (a_i, b_i) \in S\},$$

that is, the set of the indices of the gates crossing the cut S. Without loss of generality, we can assume that for each $i \in R$, we have $a_i \in A$ and $b_i \in B$.

Since in a single layer, there is at most one gate acting on a particular adjacent pair of qubits, we have

$$|R| \leq O(d\sqrt{n}).$$

Breaking the gates in R. Now, for each $i \in R$, we decompose O_i (which can be viewed as a matrix in $\mathbb{C}^{4\times 4}$) into a sum of 16 single-entry matrices $O_{i,1}, O_{i,2}, dotsc, O_{i,16}$.

Write O_i as

$$O_i = \sum_{x,y\in\{0,1\}^2} \langle y|O_i|x\rangle \cdot |y\rangle\langle x|.$$

Then we set $O_{i,j} = \langle y_j|O_i|x_j\rangle \cdot |y_j\rangle\langle x_j|$ for each $j \in [16]$, where (x_j, y_j) is the j-th ordered pair in $\{0,1\}^2 \times \{0,1\}^2$.

Decomposing the instance. Now, we are going to expand each $U_i = O_i \otimes I_{[n]\setminus\{a_i,b_i\}}$ as a sum

$$U_i = \sum_{j=1}^{16} O_{i,j} \otimes I_{[n]\setminus\{a_i,b_i\}}$$

for each $i \in R$, and therefore decompose the answer $\langle y|C|x\rangle = \langle y|U_m U_{m-1} \cdots U_1|x\rangle$ into a sum of $16^{|R|}$ terms. More concretely, for a mapping τ from R to $[16]$ and an index $i \in [m]$, we define

$$U_{i,tau} = \begin{cases} O_{i,tau(i)} \times I_{[n]\setminus\{a_i,b_i\}} & i \in R. \\ U_i & i \notin R. \end{cases}$$

Let $\mathcal{T}$ be the set of all mappings from R to $[16]$. Then we have

$$\langle y|C|x\rangle = \langle y|U_m U_{m-1} \cdots U_1|x\rangle = \sum_{\tau\in\mathcal{T}} \langle y|U_{m,tau} U_{m-1,tau} \cdots U_{1,tau}|x\rangle.$$

Dealing with the sub-instance. For each $\tau \in \mathcal{T}$ and an index $i \in [m]$, we are going to show that $U_{i,tau}$ can be decomposed as $U^A_{i,tau} \otimes U^B_{i,tau}$, where $U^A_{i,tau}$ and $U^B_{i,tau}$ are operators on $\mathcal{H}_A$ and $\mathcal{H}_B$ respectively.

When $i \in R$, by definition, there exist $x, y \in \{0,1\}^2$ and $\alpha \in \mathbb{C}$ such that

$$U_{i,tau} = \alpha \cdot |y\rangle\langle x| \otimes I_{[n]\setminus\{a_i,b_i\}} = \alpha \cdot \left(|y_0\rangle\langle x_0| \otimes I_{A\setminus\{a_i\}}\right) \otimes \left(|y_1\rangle\langle x_1| \otimes I_{B\setminus\{b_i\}}\right).$$

Otherwise $i \notin R$. In this case, if O_i is of the form $O_i \otimes I_{[n]\setminus\{a_i,b_i\}}$, then a_i, b_i must be both in A or in B and the claim trivially holds; and the claim is also obvious when O_i is of the form $O_i \otimes I_{[n]\setminus\{a_i\}}$ or $I_{[n]}$.

Moreover, one can easily verify that each $U^A_{i,tau}$ is of the form $O^A_i \otimes I_{A\setminus\{a_i,b_i\}}$ or $O^A_i \otimes I_{A\setminus\{a_i\}}$ or simply I_A, in which O^A_i is (respectively) a 2-qubit operator on $\mathcal{H}_{\{a_i,b_i\}}$ or a 1-qubit operator on $\mathcal{H}_{\{a_i\}}$), and the same holds for each $U^B_{i,tau}$.

Hence, we have

$$\begin{aligned}
&\langle y|U_m U_{m-1}\cdots U_1|x\rangle \\
&= \sum_{\tau\in\mathcal{T}} \langle y|U_{m,tau}U_{m-1,tau}\cdots U_{1,tau}|x\rangle .notag \qquad (3)\\
&= \sum_{\tau\in\mathcal{T}} \langle y|(U^A_{m,tau}\otimes U^B_{m,tau})(U^A_{m-1,tau}\otimes U^B_{m-1,tau})\cdots(U^A_{1,tau}\otimes U^B_{1,tau})|x\rangle .notag \qquad (4)\\
&= \sum_{\tau\in\mathcal{T}} \langle y_A|U^A_{m,tau}U^A_{m-1,tau}\cdots U^A_{1,tau}|x_A\rangle \cdot \langle y_B|U^B_{m,tau}U^B_{m-1,tau}\cdots U^B_{1,tau}|x_B\rangle, \qquad (5)
\end{aligned}$$

where x_A, x_B (y_A, y_B) is the projection of x (y) on $\mathcal{H}_A$ and $\mathcal{H}_B$.

So from the above discussion, we can then calculate $\langle y_A|U^A_{m,tau}U^A_{m-1,tau}\cdots U^A_{1,tau}|x_A\rangle$ with a recursive call with computational basis states $|x_A\rangle$ and $|y_A\rangle$, grid A, and m gates $U^A_{1,tau}, U^A_{2,tau}, dotsc, U^A_{m,tau}$.

The matrix element $\langle y_B|U^B_{m,tau}U^B_{m-1,tau}\cdots U^B_{1,tau}|x_B\rangle$ can be computed similarly. After that we sum up all the terms in (5) to get the answer.

Complexity analysis. Now we are going to bound the running time. Let $F(n)$ be an upper bound on the running time when the size of the remaining grid is n. Then we have

$$F(n) = \begin{cases} O(m) & \text{when } n = 1. \\ 2^{O(d\sqrt{n})} \cdot \max_{k\in[n/3,2n/3]} F(k) & \text{otherwise.} \end{cases}$$

The second case is due to the fact that the sizes of sub-instances (i.e., the sizes of A and B) lie in $[n/3, 2n/3]$, and $\mathcal{T} = 16^{|R|} = 2^{O(d\sqrt{n})}$. It is not hard to see that $F(n)$ is an increasing function, so we have $F(n) = 2^{O(d\sqrt{n})}F(2n/3)$ for $n > 1$, which further simplifies to $F(n) = 2^{O(d\sqrt{n})}$.

Finally, we can see that at each recursion level, we need $O(d \cdot n)$ space to store the circuit, and $O(1)$ space to store the intermediate answer. Since there are at most $\log n$ recursion levels, the space complexity is $O(d \cdot n \log n)$. ◀

Interestingly, by using tensor network methods, Markov and Shi [43] gave an algorithm for simulating quantum circuits on grids with similar running time to ours. However, the difference is that Markov and Shi's algorithm requires $2^{O(d\sqrt{n})}$ time *and* $2^{O(d\sqrt{n})}$ space, whereas ours requires $2^{O(d\sqrt{n})}$ time and only polynomial space.

The algorithm of Theorem 17 achieves a speedup over Theorem 15 only for small d, but we can combine it with the algorithm in Theorem 15 to get a faster algorithm for the whole range of d.

▶ **Theorem 18.** *There is a constant c such that, given a quantum circuit C on n qubits with depth d, and two computational basis states $|x\rangle, spzy$, assuming that G_C can be embedded into a two dimensional grid with size n (with the embedding explicitly specified), we can compute $\langle y|C|x\rangle$ in*

$$O(2^n \cdot \left[1 + \left(\frac{d}{c\sqrt{n}}\right)^{n+1}\right])$$

time and $O(d \cdot n \log n)$ space.

Proof. By Theorem 17, there is a constant c such that we have an $O(2^n)$ time and polynomial space algorithm for calculating $\langle y|C|x\rangle$ when the depth is at most $c\sqrt{n}$ for circuit on grids.

So we can use the same algorithm as in Theorem 15, except that we revert to the algorithm in Theorem 17 when the depth is no more than $c\sqrt{n}$.

We still let $F(d)$ be the running time on a circuit of d layers. We then have $F(d) = O(2^n)$ when $d \leq c\sqrt{n}$. From the above discussion, we can see that for $d > c\sqrt{n}$,

$$F(d) \leq 2^{n+1} \cdot F(\lceil \frac{d}{2} \rceil) = O(2^n \cdot 2^{(n+1)\lceil \log(d/c\sqrt{n}) \rceil}) = O(2^n \cdot \left(\frac{d}{c\sqrt{n}}\right)^{n+1}),$$

which proves the running time bound. And it is not hard to see that the algorithm's space usage is dominated by $O(d \cdot n \log n)$. ◀

4.3 Space-Time Trade-off Schemes

We now show how to optimize the running time for whatever space is available.

▶ **Theorem 19.** *Given a quantum circuit C on n qubits with depth d, two computational basis states $|x\rangle, spzy$ and an integer k, we can compute $\langle y|C|x\rangle$ in*

$$O(n2^{n-k} \cdot 2^{(k+1)\lceil \log d \rceil}) \leq O(n2^{n-k} \cdot (2d)^{k+1})$$

time and $O(2^{n-k} \log d)$ space.

Proof.

Decomposing the whole Hilbert space $\mathcal{H}_{[n]}$. We first decompose $\mathcal{H}_{[n]}$ into a direct sum of many subspaces. Let w_i be the i-th string in $\{0,1\}^k$ in lexicographic order. For each $i \in [2^k]$, let $\mathcal{H}_i = \mathsf{Span}(|w_i 0^{n-k}\rangle, ...|w_i 1^{n-k}\rangle)$. Then we have

$$\mathcal{H}_{[n]} = \bigoplus_{i=1}^{2^k} \mathcal{H}_i.$$

Also, let $\mathcal{P}_i$ be the projection from $\mathcal{H}_{[n]}$ to $\mathcal{H}_i$; then

$$I_{[n]} = \sum_{i=1}^{2^k} \mathcal{P}_i.$$

Now we generalize the original problem as follows: given two indices $s, t \in [2^k]$ and a pure state $|u\rangle$ in $\mathcal{H}_s$, we want to compute $\mathcal{P}_t C|u\rangle$. By choosing s and t such that $\mathcal{H}_s$ contains $|x\rangle$ and $\mathcal{H}_t$ contains $|y\rangle$, we can easily solve the original problem.

The base case $d = 1$. When there is only one layer, $\mathcal{P}_t C|u\rangle$ can be calculated straightforwardly in $O(n \cdot 2^{n-k})$ time and $O(2^{n-k})$ space.

Recursion. When $d > 1$, we have

$$\begin{aligned}
\mathcal{P}_t C|u\rangle &= \mathcal{P}_t C_{[d \leftarrow d/2+1]} \cdot C_{[d/2 \leftarrow 1]}|u\rangle \\
&= \mathcal{P}_t C_{[d \leftarrow d/2+1]} \left(\sum_{z \in [2^k]} \mathcal{P}_z \right) C_{[d/2 \leftarrow 1]}|u\rangle \\
&= \sum_{z \in [2^k]} \mathcal{P}_t C_{[d \leftarrow d/2+1]} \mathcal{P}_z C_{[d/2 \leftarrow 1]}|u\rangle.
\end{aligned}$$

We can then calculate $\mathcal{P}_t C_{[d \leftarrow d/2+1]} \mathcal{P}_z C_{[d/2 \leftarrow 1]}|u\rangle$ for each z as follows: we first use a recursive call to get $|b\rangle = \mathcal{P}_z C_{[d/2 \leftarrow 1]}|u\rangle$ and a second recursive call to compute $\mathcal{P}_t C_{[d \leftarrow d/2+1]}|b\rangle$ (note that $|b\rangle \in \mathcal{H}_z$).

Complexity analysis. It is easy to see that the total space usage is $O(2^{n-k}\log d)$, since for each i, storing a vector in $\mathcal{H}_i$ takes $O(2^{n-k})$ space, and we only need to record $O(1)$ such vectors at each recursion level. In addition, when $d=1$, we need only $O(2^{n-k})$ space.

For the running time bound, let $F(d)$ denote the running time on a circuit of d layers; then $F(1)=O(n2^{n-k})$. From the above discussion, it follows that

$$F(d)\leq 2^{k+1}\cdot F(\lceil d/2\rceil)=O(n2^{n-k}\cdot 2^{(k+1)\lceil\log(d)\rceil})=O(n2^{n-k}\cdot(2d)^{k+1}).$$ ◀

The above trade-off scheme can be further improved for quantum circuits on grids.

▶ **Theorem 20.** *There is a constant c such that, given a quantum circuit C on n qubits with depth d, two computational basis states $|x\rangle, spzy$ and an integer k, assuming that G_C can be embedded into a two dimensional grid with size n, we can compute $\langle y|C|x\rangle$ in*

$$2^{O(n)}\cdot\left[1+(2d/c\sqrt{n})^{k+1}\right]$$

time and

$$O\big(2^{n-k}\max(1,log(d/\sqrt{n}))\big)$$

space.

Proof. By Theorem 17, there is a constant c such that we have an $O(2^n)$ time algorithm for calculating $\langle y|C|x\rangle$ for circuits on grids with depth at most $c\sqrt{n}$.

Then we use the same algorithm as in Theorem 19, with the only modification that when $d\leq c\sqrt{n}$, we calculate $\mathcal{P}_t\cdot C|u\rangle$ by $2^{2(n-k)}$ calls of the algorithm in Theorem 17.

With the same analysis as in Theorem 19, when $d>c\sqrt{n}$, we can see that the total space usage is $O(2^{n-k}\log(d/c\sqrt{n}))$, and the running time is

$$O(2^{n+2(n-k)+(k+1)\lceil\log(d/c\sqrt{n})\rceil})=O(2^{O(n)}\cdot(2d/c\sqrt{n}))^{k+1}).$$

Combining with the algorithm for $d\leq c\sqrt{n}$ proves our running time and space bound. ◀

5 Strong Quantum Supremacy Theorems Must Be Non-Relativizing

In this section we show that there is an oracle relative to which SampBPP = SampBQP, yet $\mathsf{PH}^{\mathcal{O}}$ is infinite.

Recall that an oracle $\mathcal{O}$ is a function $\mathcal{O}:\{0,1\}^*\to\{0,1\}$, and the combination of two oracles $\mathcal{O}_0, oracle_1$, denoted as $\mathcal{O}_0\oplus\mathcal{O}_1$, simply maps $z\in\{0,1\}^*$ to $\mathcal{O}_{z_1}(z_2,z_3,dotsc,z_{|z|})$ (cf. citefenner2003oracle). We use $\mathcal{O}_n$ to denote the restriction of $\mathcal{O}$ on $\{0,1\}^n$.

5.1 Intuition

We have two simultaneous objectives: (1) we need SampBPP and SampBQP to be equal; and (2) we also need PH to be infinite. So it will be helpful to review some previous results on (1) and (2) separately.

- An oracle $\mathcal{O}$ such that $\mathsf{SampBPP}^{\mathcal{O}}=\mathsf{SampBQP}^{\mathcal{O}}$: in order to make two classes equal, we can use the standard method: *adding a much more powerful oracle* [16]. That is, we set $\mathcal{O}$ to be a PSPACE-complete language, like TQBF. Then it is easy to see both $\mathsf{SampBPP}^{\mathsf{TQBF}}$ and $\mathsf{SampBQP}^{\mathsf{TQBF}}$ become SampPSPACE (i.e., the class of approximate sampling problems solvable in polynomial space).
- An oracle $\mathcal{O}$ such that $\mathsf{PH}^{\mathcal{O}}$ is infinite: a line of works by Yao [60], Håstad [35], and others constructed relativized worlds where PH is infinite, and a very recent breakthrough by Rossman, Servedio, and Tan [49] even shows that PH is infinite relative to a random oracle with probability 1.

A Failed Attempt: Direct Combination

The first natural idea is to combine the previous two results straightforwardly by setting the oracle to be $\mathsf{TQBF} \oplus \mathcal{O}$, where $\mathcal{O}$ is a random oracle.

Alas, it is not hard to see that this does not work: while PH is still infinite, a SampBQP algorithm can perform Fourier Sampling (cf. Definition 27) on the random oracle bits, and it is known that no SampBPP algorithm can do that [9] (see also Theorem 33). Hence, in this case $\mathsf{SampBQP} \neq \mathsf{SampBPP}$.

Another Failed Attempt: Hiding a "Secret Random String" in a Secret Location

The failure of the naive approach suggests that we must somehow "hide" the random oracle bits, since if the SampBQP algorithm has access to them, then SampBPP and SampBQP will not be equal. More specifically, we want to hide a "secret random string" among the oracle bits so that:

1. a PH algorithm can find it, so that PH is still infinite, but
2. a SampBQP algorithm cannot find it, so that we can still make $\mathsf{SampBPP} = \mathsf{SampBQP}$ by attaching a TQBF oracle.

Inspired by the so-called cheat-sheet construction [10], it is natural to consider a direct hiding scheme. Imagine that the oracle bits are partitioned into two parts: one part is $\log N$ copies of the OR function on N bits, and another part is N binary strings $y_1, dotsc, y_N$, each with length N. Let $t = a_1, a_2, dotsc, a_{\log N} \in \{0,1\}^{\log N}$ be the answer to the copies of OR; we can also interpret t as an integer in $[N]$. Finally, set y_t to be a random input, while other y_i's are set to zero.

Intuitively, a PH algorithm can easily evaluate the $\log N$ copies of OR and then get access to the random string; while it is known that OR is hard for a quantum algorithm, so no quantum algorithm should be able to find the location of the random string efficiently.

Unfortunately, there is a fatal issue with the above approach: a SampBQP algorithm is also given an input $x \in \{0,1\}^n$ and it may *guess* that the input x denotes the location of the random string. That is, on some particular input, the SampBQP algorithm is "lucky" and gets access to the random string, which still makes SampBPP and SampBQP unequal.

Hiding the "Secret Random String" in a Bunch of OR's

Therefore, our final construction goes further. Instead of hiding the random string in a secret location amid the oracle bits, we hide it using a bunch of ORs. That is, suppose we want to provide N uniform random bits. Then we provide them each as an OR of N bits. In this way, a PH algorithm is still able to access the random bits, while a quantum algorithm, even if it's "lucky" with its additional input, still can't get access to these hidden random bits.

5.2 Implementation

The Distribution $\mathcal{D}_{\mathcal{O}}$ on Oracles

We first describe formally how to hide a random string inside a bunch of OR's by defining a distribution $\mathcal{D}_{\mathcal{O}}$ on oracles.

For notational convenience, our constructed oracles always map all odd-length binary strings to 0. So we can alternatively describe such an oracle $\mathcal{O}$ by a collection of functions $\{f_n\}_{n=0}^{+\infty}$, where each f_n is a function from $\{0,1\}^{2n} \to \{0,1\}$. That is, $\mathcal{O}_{2n}$ is set to be f_n for each n, while the $\mathcal{O}_{2n+1}$'s are all constant zero functions.

For each string $p \in \{0,1\}^n$, we use $B_{n,p}$ to denote the set of strings in $\{0,1\}^{2n}$ with p as a prefix. Now we first define a distribution $\mathcal{D}_n$ on functions $\{0,1\}^{2n} \to \{0,1\}$, from which a sample function f_n is generated as follows: initially, we set $f_n(x) = 0$ for all $x \in \{0,1\}^{2n}$; then for each $p \in \{0,1\}^n$, with probability 0.5, we pick an element e in $B_{n,p}$ at uniformly random and set $f_n(e) = 1$. Observe that by taking the OR of each $B_{n,p}$, we get a function $g(p) := \vee_{x \in B_{n,p}} f_n(x)$, which is a uniform random function from $\{0,1\}^n$ to $\{0,1\}$ by construction.

Finally, the $\mathcal{D}_n$'s induce a distribution $\mathcal{D}_{\mathcal{O}}$ on oracles, which generates an oracle $\mathcal{O}$ by drawing $f_n \sim \mathcal{D}_n$ independently for each integer n. That is, we set $\mathcal{O}_{2n}$ to be f_n, and $\mathcal{O}_{2n+1}$ to be $\mathbf{0}$, for each n.

Having defined the distribution $\mathcal{D}_{\mathcal{O}}$, we are ready to state our result formally.

▶ **Theorem 21.** *For an oracle $\mathcal{O}$ drawn from the distribution $\mathcal{D}_{\mathcal{O}}$, the following two statements hold with probability 1:*

- $\mathsf{SampBPP}^{\mathsf{TQBF},oracle} = \mathsf{SampBQP}^{\mathsf{TQBF},oracle}$.
- $\mathsf{PH}^{\mathsf{TQBF},oracle}$ *is infinite.*

From which our desired result follows immediately.

▶ **Corollary 22.** *There exists an oracle $\mathcal{O}' = TQBF \oplus \mathcal{O}$ such that $\mathsf{SampBPP}^{\mathcal{O}'} = \mathsf{SampBQP}^{\mathcal{O}'}$ and $\mathsf{PH}^{\mathcal{O}'}$ is infinite.*

The rest of this section is devoted to the proof of Theorem 21.

5.3 $\mathsf{SampBPP}^{\mathsf{TQBF},oracle} = \mathsf{SampBQP}^{\mathsf{TQBF},oracle}$ with Probability 1

We first describe an algorithm for simulating $\mathsf{SampBQP}^{\mathsf{TQBF},oracle}$ in $\mathsf{SampBPP}^{\mathsf{TQBF},oracle}$, thereby proving the first part of Theorem 21. In the following, we assume that all oracle algorithms are given access to two oracles, TQBF and $\mathcal{O}$.

Given a SampBQP oracle algorithm M, our central task is to give a SampBPP oracle algorithm that simulates M closely. Formally:

▶ **Lemma 23.** *For any* SampBQP *oracle algorithm M, there is a* SampBPP *oracle algorithm A such that:*

Let $\mathcal{O}$ be an oracle drawn from $\mathcal{D}_{\mathcal{O}}$, and let $\mathcal{D}^{M}_{x,varepsilon}$ and $\mathcal{D}^{A}_{x,varepsilon}$ be the distributions output by $M^{TQBF,oracle}$ and $A^{TQBF,oracle}$ respectively on input $\langle x, 0^{1/\varepsilon}\rangle$. Then with probability at least $1 - \exp\{-(2 \cdot |x| + 1/\varepsilon)\}$, we have

$$\|\mathcal{D}^{M}_{x,varepsilon} - \mathcal{D}^{A}_{x,varepsilon}\| \leq \varepsilon.$$

Before proving Lemma 23, we show it implies the first part of Theorem 21.

Proof of the first part of Theorem 21. Fix a SampBQP oracle algorithm M, and let $\mathcal{O}$ be an oracle drawn from $\mathcal{D}_{\mathcal{O}}$. We first show that with probability 1, there is a classical algorithm A_M such that

$$\|\mathcal{D}^{M}_{x,varepsilon} - \mathcal{D}^{A_M}_{x,varepsilon}\| \leq \varepsilon \text{ for all } x \in \{0,1\}^* \text{ and } \varepsilon = 2^{-k} \text{ for some integer } k. \quad (6)$$

Let A be the SampBPP algorithm guaranteed by Lemma 23. For an input $x \in \{0,1\}^*$ and an integer k, we call (x,k) a *bad pair* if $\|\mathcal{D}^{M}_{x,2^{-k}} - \mathcal{D}^{A}_{x,2^{-k}}\| > 2^{-k}$. By Lemma 23, the expected number of bad pairs is upper-bounded by

$$\sum_{n=1}^{+\infty} 2^n \cdot \sum_{k=1}^{+\infty} \exp(-(2n + 2^k)) \leq \sum_{n=1}^{+\infty} \sum_{k=1}^{+\infty} \exp(-(n+k)) \leq O(1).$$

This means that with probability 1, there are only finitely many bad pairs, so we can handle them by hardwiring their results into the algorithm A to get the algorithm A_M we want.

Since there are only countably many SampBQP oracle algorithms M, we see with probability 1, for every SampBQP oracle algorithm M, there is a classical algorithm A_M such that (6) holds. We claim that in that case, $\mathsf{SampBQP}^{\mathsf{TQBF},oracle} = \mathsf{SampBPP}^{\mathsf{TQBF},oracle}$.

Let $\mathcal{S}$ be a sampling problem in $\mathsf{SampBQP}^{\mathsf{TQBF},oracle}$. This means that there is a SampBQP oracle algorithm M, such that for all $x \in \{0,1\}^*$ and ε, we have $\|\mathcal{D}^M_{x,varepsilon} - \mathcal{S}_x\| \leq \varepsilon$. Let A_M be the corresponding SampBPP algorithm. Now consider the following algorithm A': given input $\langle x, 0^{1/\varepsilon}\rangle$, let k be the smallest integer such that $2^{-k} \leq \varepsilon/2$; then run A_M on input $\langle x, 0^{2^k}\rangle$ to get a sample from $\mathcal{D}^{A_M}_{x,2^{-k}}$.

Since

$$\begin{aligned}\|\mathcal{D}^{A'}_{x,varepsilon} - S_x\| &= \|\mathcal{D}^{A_M}_{x,2^{-k}} - S_x\| \\ &\leq \|\mathcal{D}^{M}_{x,2^{-k}} - \mathcal{D}^{A_M}_{x,2^{-k}}\| + \|\mathcal{D}^{M}_{x,2^{-k}} - \mathcal{S}_x\| \leq 2 \cdot 2^{-k} \leq \varepsilon,\end{aligned}$$

this means that A' solves $\mathcal{S}$ and $\mathcal{S} \in \mathsf{SampBPP}^{\mathsf{TQBF},oracle}$. So $\mathsf{SampBQP}^{\mathsf{TQBF},oracle} \subseteq \mathsf{SampBPP}^{\mathsf{TQBF},oracle}$ with probability 1, which completes the proof. ◀

We now prove Lemma 23, which is the most technical part of the whole section.

Proof of Lemma 23. Recall that from the canonical description in Section 2.2, there exists a fixed polynomial p, such that given input $\langle x, 0^{1/\varepsilon}\rangle$, the machine M first constructs a quantum circuit C with $N = p(|x|, 1/\varepsilon)$ qubits and N gates classically (C can contain TQBF and $\mathcal{O}$ gates). We first set up some notation.

Notation. Recall that $\mathcal{O}$ can be specified by a collection of functions $\{f_n\}_{n=0}^{+\infty}$, where each f_n maps $\{0,1\}^{2n}$ to $\{0,1\}$. Without loss of generality, we can assume that all the $\mathcal{O}$ gates act on an even number of qubits, and for each n, all the f_n gates act on the first $2n$ qubits.

For a function $f : \{0,1\}^k \to \{0,1\}$, we use U_f to denote the unitary operator mapping $|i\rangle$ to $(-1)^{f(i)}|i\rangle$ for $i \in \{0,1\}^k$.

Suppose there are T $\mathcal{O}$-gates in total, and suppose the i-th $\mathcal{O}$-gate is an f_{n_i} gate. Then the unitary operator U applied by the circuit C can be decomposed as

$$U = U_{T+1}(U_{f_{n_T}} \otimes I_{N-2n_T}) \cdots (U_{f_{n_2}} \otimes I_{N-2n_2})U_2(U_{f_{n_1}} \otimes I_{N-2n_1})U_1,$$

where the U_i's are the unitary operators corresponding to the sub-circuits which don't contain an $\mathcal{O}$ gate.

Our algorithm proceeds by replacing each $\mathcal{O}$-gate by a much simpler gate, one by one, without affecting the final quantum state too much. It then simulates the final circuit with the help of the TQBF oracle.

Replacing the t-th $\mathcal{O}$-gate. Suppose we have already replaced the first $t-1$ $\mathcal{O}$-gates. That is, for each $i \in [t-1]$, we replaced the f_{n_i} gate (the i-th $\mathcal{O}$-gate) with a g_i gate, and now we are going to replace the t-th $\mathcal{O}$-gate.

Let

$$|v\rangle = U_t(U_{g_{t-1}} \otimes I_{N-2n_{t-1}}) \cdots (U_{g_2} \otimes I_{N-2n_2})U_2(U_{g_1} \otimes I_{N-2n_1})U_1|0\rangle^{\otimes N},$$

which is the quantum state right before the t-th $\mathcal{O}$ gate in the circuit after the replacement.

For brevity, we use f to denote the function f_{n_t}, and we drop the subscript t of n_t when it is clear from context.

Analysis of incurred error. The t-th $\mathcal{O}$-gate is an f gate. If we replace it by a g gate, the change to the quantum state is

$$\|U_f \otimes I_{N-2n}|v\rangle - U_g \otimes I_{N-2n}|v\rangle\| = \|(U_f - U_g) \otimes I_{N-2n}|v\rangle\|.$$

We can analyze the above deviation by bounding its square. Let H be the Hilbert space spanned by the last $N-2n$ qubits, and let $\rho = \mathrm{Tr}_H[|v\rangle\langle v|]$. Then we have

$$\begin{aligned}
&\|((U_f - U_g) \otimes I_{N-2n})|v\rangle\|^2 \\
=&\,\mathrm{Tr}\Big[(U_f - U_g)^\dagger (U_f - U_g) \otimes I_{N-2n}|v\rangle\langle v|\Big] \\
=&\,\mathrm{Tr}\Big[(U_f - U_g)^\dagger (U_f - U_g)\rho\Big].
\end{aligned}$$

Note that

$$(U_f - U_g)^\dagger (U_f - U_g) = 4 \sum_{f(i)\neq g(i)} |i\rangle\langle i|$$

from the definition. So we can further simplify the above trace as

$$\mathrm{Tr}\Big[(U_f - U_g)^\dagger (U_f - U_g)\rho\Big] = 4 \sum_{f(i)\neq g(i)} \mathrm{Tr}[|i\rangle\langle i|\rho] = 4 \sum_{f(i)\neq g(i)} \langle i|\rho|i\rangle. \tag{7}$$

Now, ρ is a (mixed) quantum state on the first $2n$ bits, and $\langle i|\rho|i\rangle$ is the probability of seeing i when measuring ρ in the computational basis. So we can define a probability distribution Q on $\{0,1\}^{2n}$ by $Q(i) := \langle i|\rho|i\rangle$.

Using the distribution Q, the error term (7) can finally be simplified as:

$$4 \sum_{i\in\{0,1\}^{2n}} Q(i) \cdot [f(i) \neq g(i)] = 4 \cdot \Pr_{i\sim Q}[f(i) \neq g(i)], \tag{8}$$

where $[f(i) \neq g(i)]$ is the indicator function that takes value 1 when $f(i) \neq g(i)$ and 0 otherwise.

A posterior distribution $\mathcal{D}_n^{\mathsf{post}}$ on functions from $\{0,1\}^{2n} \to \{0,1\}$. Now, recall that $f = f_n$ is a function drawn from the distribution $\mathcal{D}_n$. Our goal is to replace f by another simple function g, such that with high probability, the introduced deviation (8) is small.

Note that when replacing the t-th $\mathcal{O}$ gate, we may already have previously queried some contents of f (i.e., it is not the first f_n gate in the circuit). So we need to consider the posterior distribution $\mathcal{D}_n^{\mathsf{post}}$ on functions from $\{0,1\}^{2n} \to \{0,1\}$. That is, we want a function g, such that with high probability over $f \sim \mathcal{D}_n^{\mathsf{post}}$, the error term (8) is small.

We use a function $f_{\mathsf{known}} : \{0,1\}^{2n} \to \{0,1,*\}$ to encode our knowledge: if $f(i)$ is not queried, then we set $f_{\mathsf{known}}(i) := *$; otherwise we set $f_{\mathsf{known}}(i) := f(i)$. Then $\mathcal{D}_n^{\mathsf{post}}$ is simply the distribution obtained from $\mathcal{D}_n$ by conditioning on the event that f is consistent with f_{known}.

We can now work out the posterior distribution $\mathcal{D}_n^{\mathsf{post}}$ from the definition of $\mathcal{D}_n$ and Bayes' rule.

For $f \sim \mathcal{D}_n^{\mathsf{post}}$, we can see that all the sets $B_{n,p}$ (recall that $B_{n,p}$ is the set of all strings in $\{0,1\}^{2n}$ with p as a prefix) are still independent. So we can consider each set separately.

For each $p \in \{0,1\}^n$, if there is an $x \in B_{n,p}$ such that $f_{\mathsf{known}}(x) = 1$, then by the construction of $\mathcal{D}_n$, all other elements $y \in B_{n,p}$ must satisfy $f(y) = 0$.

Otherwise, if there is no $x \in B_{n,p}$ such that $f_{\mathsf{known}}(x) = 1$, then we set $Z_p = |\{f_{\mathsf{known}}(x) = 0 \mid x \in B_{n,p}\}|$ and note that $|B_{n,p}| = 2^n$. By Bayes' rule, we see that with probability $\frac{1}{2 - Z_p \cdot 2^{-n}}$, all $y \in B_{n,p}$ satisfy $f(y) = 0$; and for each $y \in B_{n,p}$ such that $f_{\mathsf{known}}(y) = *$, with probability $\frac{2^{-n}}{2 - Z_p \cdot 2^{-n}}$, we have that y is the only element of $B_{n,p}$ that satisfies $f(y) = 1$.

Construction and Analysis of g. Our construction of g goes as follows: we first set $g(x) = f_{\mathsf{known}}(x)$ for all x such that $f_{\mathsf{known}}(x) \neq *$. Then for a parameter τ which will be specified later, we query all $x \in \{0,1\}^{2n}$ with $Q(x) \geq \tau$, and set $g(x) = f(x)$ for them. For all other positions of g, we simply set them to zero. Hence, there are at most $O(1/\tau) + W$ ones in g, where W denotes the number of ones in f_{known}.

The following three properties of g are immediate from the construction.

- $f(x) \neq g(x)$ implies $Q(x) \leq \tau$. (9)
- $g(x) = 1$ implies $f(x) = g(x)$. (10)
- For each $p \in \{0,1\}^n$, there is at most one $x \in B_{n,p}$ with $f(x) \neq g(x)$. (11)

Upper bounding the deviation (8). Now we are going to show that $\Pr_{x \sim Q}[f(x) \neq g(x)]$ is very small, with overwhelming probability over the posterior distribution $\mathcal{D}_n^{\mathsf{post}}$.

We first define 2^n random variables $\{X_p\}_{p \in \{0,1\}^n}$, where $X_p = \sum_{x \in B_{n,p}} Q(x) \cdot [f(x) \neq g(x)]$ for each $p \in \{0,1\}^n$. By the construction of $\mathcal{D}_n^{\mathsf{post}}$, we can see that all X_p's are independent. Moreover, by properties (9) and (11), there is at most one $x \in B_{n,p}$ such that $f(x) \neq g(x)$, and that x must satisfy $Q(x) \leq \tau$. Therefore $X_p \in [0, tau]$ for every p.

Let $X = \sum_{p \in \{0,1\}^n} X_p$, and $\mu = \mathbb{E}[X]$. Alternatively, we can write X as

$$X = \sum_{x \in \{0,1\}^{2n}} Q(x) \cdot [f(x) \neq g(x)],$$

so

$$\mu = \sum_{x \in \{0,1\}^{2n}} Q(x) \cdot \mathbb{E}[f(x) \neq g(x)].$$

We claim that $\mathbb{E}[f(x) \neq g(x)] \leq 2^{-n}$ for all $x \in \{0,1\}^{2n}$, and consequently $\mu \leq 2^{-n}$. Fix an $x \in \{0,1\}^{2n}$, and suppose $x \in B_{n,p}$. When $g(x) = 1$, we must have $f(x) = g(x)$ by property (10). When $g(x) = 0$, by the definition of $\mathcal{D}_n^{\mathsf{post}}$, we have $f(x) = 1$ with probability at most $\frac{2^{-n}}{2 - Z_p \cdot 2^{-n}} \leq 2^{-n}$. So $\mathbb{E}[f(i) \neq g(i)] \leq 2^{-n}$ in both cases and the claim is established.

Applying the Chernoff Bound. Set $\delta = \frac{\mu^{-1}\varepsilon^4}{32T^2}$. If $\delta \leq 1$, then we have

$$32T^2\varepsilon^{-4} \geq \mu^{-1} \geq 2^n.$$

This means that we can simply query all the positions in f_n using $2^{2n} = O(T^4 \cdot \varepsilon^{-8})$ queries, as this bound is polynomial in $|x|$ and $1/\varepsilon$ (recall that $T \leq N = p(|x|, 1/\varepsilon)$).

Hence, we can assume that $\delta > 1$. So by Corollary 7, we have

$$\Pr[X \geq 2\delta\mu] \leq \Pr[X \geq (1+\delta)\mu] \leq \exp\left\{-\frac{\delta\mu}{3\tau}\right\}.$$

Finally, we set $\tau = \dfrac{\varepsilon^4}{96T^2 \cdot (2n + \varepsilon^{-1} + \ln T)}$.

Therefore, with probability

$$1 - \exp\left\{-\frac{\delta\mu}{3\tau}\right\} = 1 - \exp(-(2n + \varepsilon^{-1} + \ln T)) = 1 - \frac{\exp(-(2n+\varepsilon^{-1})}{T},$$

we have

$$\|(U_f - U_g) \otimes I_{N-2n}|v\rangle\|^2 = 4 \cdot X \leq 8\delta\mu = \frac{\varepsilon^4}{4T^2},$$

which in turn implies

$$\|(U_f - U_g) \otimes I_{N-2n}|v\rangle\| \leq \frac{\varepsilon^2}{2T}.$$

Moreover, we can verify that g only has $O(1/\tau) + W = \mathrm{poly}(n, 1/\varepsilon)$ ones.

Analysis of the final circuit C^{final}. Suppose that at the end, for each $t \in [T]$, our algorithm has replaced the t-th $\mathcal{O}$-gate with a g_t gate, where g_t is a function from $\{0,1\}^{2n_t}$ to $\{0,1\}$. Let C^{final} be the circuit after the replacement.

Let

$$V = U_{T+1}(U_{g_T} \otimes I_{N-2n_T}) \cdots (U_{g_2} \otimes I_{N-2n_2})U_2(U_{g_1} \otimes I_{N-2n_1})U_1$$

be the unitary operator corresponding to C^{final}. Also, recall that U is the unitary operator corresponding to the original circuit C. We are going to show that $U|0\rangle^{\otimes N}$ and $V|0\rangle^{\otimes N}$, the final quantum states produced by U and V respectively, are very close.

We first define a sequence of intermediate quantum states. Let $|u_1\rangle = U_1|0\rangle^{\otimes N}$. Then for each $t > 1$, we define

$$|u_t\rangle = U_t(U_{f_{n_{t-1}}} \otimes I_{N-2n_{t-1}})|u_{t-1}\rangle.$$

That is, $|u_t\rangle$ is the quantum state immediately before applying the t-th $\mathcal{O}$-gate in the original circuit. Similarly, we let $|v_1\rangle = U_1|0\rangle^{\otimes N}$, and

$$|v_t\rangle = U_t(U_{g_{t-1}} \otimes I_{N-2n_{t-1}})|u_{t-1}\rangle$$

for each $t > 1$.

From the analysis of our algorithm, over $\mathcal{O} \sim \mathcal{D}_{\mathcal{O}}$, for each $t \in [T]$, with probability $1 - \exp(-(2n + \varepsilon^{-1}))/T$, we have

$$\|U_{f_{n_t}} \otimes I_{N-2n_t}|v_t\rangle - U_{g_t} \otimes I_{N-2n_t}|v_t\rangle\| \leq \frac{\varepsilon^2}{2T}. \tag{12}$$

So by a simple union bound, with probability at least $1 - \exp(-(2n + \varepsilon^{-1}))$, the above bound holds for all $t \in [T]$. We claim that in this case, for each $t \in [T+1]$, we have

$$\||v_t\rangle - |u_t\rangle\| \leq (t-1) \cdot \frac{\varepsilon^2}{2T}. \tag{13}$$

We prove this by induction. Clearly it is true for $t = 1$. When $t > 1$, suppose (13) holds for $t - 1$; then

$$\begin{aligned}
\||v_t\rangle - |u_t\rangle\| =& \|U_t(\mathcal{O}_{g_{t-1}} \otimes I_{N-2n_{t-1}})|v_{t-1}\rangle - U_t(f_{n_{t-1}} \otimes I_{N-2n_{t-1}})|u_{t-1}\rangle\| \\
=& \|U_{g_{t-1}} \otimes I_{N-2n_{t-1}}|v_{t-1}\rangle - U_{f_{n_{t-1}}} \otimes I_{N-2n_{t-1}}|u_{t-1}\rangle\| \\
\leq& \|U_{g_{t-1}} \otimes I_{N-2n_{t-1}}|v_{t-1}\rangle - U_{f_{n_{t-1}}} \otimes I_{N-2n_{t-1}}|v_{t-1}\rangle\| \\
&+ \|U_{f_{n_{t-1}}} \otimes I_{N-2n_{t-1}}|v_{t-1}\rangle - U_{f_{n_{t-1}}} \otimes I_{N-2n_{t-1}}|u_{t-1}\rangle\| \\
\leq& \frac{\varepsilon^2}{2T} + \||u_{t-1}\rangle - |v_{t-1}\rangle\| \leq (t-1) \cdot \frac{\varepsilon^2}{2T},
\end{aligned}$$

where the second line holds by the fact that U_t is unitary, the third line holds by the triangle inequality, and the last line holds by (12) and the induction hypothesis.

Upper-bounding the error. Therefore, with probability at least $1 - \exp(-(2n + \varepsilon^{-1}))$, we have

$$\||v_{T+1}\rangle - |u_{T+1}\rangle\| = \|U|0\rangle^{\otimes N} - V|0\rangle^{\otimes N}\| \leq \frac{\varepsilon^2}{2}.$$

Now, our classical algorithm A then simulates stage 2 and 3 of the SampBQP algorithm M straightforwardly. That is, it first takes a sample z by measuring $|v_{T+1}\rangle$ in the computational basis, and then outputs $A^{\mathsf{output}}(z)$ as its sample, where A^{output} is the classical algorithm used by M in stage 3.

From our previous analysis, A queries the oracle only $\mathrm{poly}(n, 1/\varepsilon)$ times. In addition, it is not hard to see that all the computations can be done in PSPACE, and therefore can be implemented in $\mathrm{poly}(n, 1/\varepsilon)$ time with the help of the TQBF oracle. So A is a SampBPP algorithm.

By Corollary 5, with probability at least $1-\exp(-(2n+\varepsilon^{-1}))$, the distribution $\mathcal{D}^A_{x,varepsilon}$ outputted by A satisfies

$$\|\mathcal{D}^A_{x,varepsilon} - \mathcal{D}^M_{x,varepsilon}\| \leq \sqrt{2 \cdot \frac{\varepsilon^2}{2}} = \varepsilon,$$

and this completes the proof of Lemma 23. ◀

5.4 $\mathsf{PH}^{\mathsf{TQBF},oracle}$ is Infinite with Probability 1

For the second part of Theorem 21, we resort to the well-known connection between PH and constant-depth circuit lower bounds.

The Average Case Constant-depth Circuit Lower Bound

For convenience, we will use the recent breakthrough result by Rossman, Servedio, and Tan [49], which shows that PH is infinite relative to a random oracle with probability 1. (Earlier constructions of oracles making PH infinite would also have worked for us, but a random oracle is a particularly nice choice.)

▶ **Theorem 24.** *Let $2 \leq d \leq \frac{c\sqrt{\log n}}{\log\log n}$, where $c > 0$ is an absolute constant. Let Sipser_d be the explicit n-variable read-once monotone depth-d formula described in [49]. Then any circuit C' of depth at most $d-1$ and size at most $S = 2^{n^{\frac{1}{6(d-1)}}}$ over $\{0,1\}^n$ agrees with Sipser_d on at most $(\frac{1}{2} + n^{-\Omega(1/d)}) \cdot 2^n$ inputs.*

$\mathcal{D}_n$ as a Distribution on $\{0,1\}^{2^{2n}}$

In order to use the above result to prove the second part of Theorem 21, we need to interpret $\mathcal{D}_n$ (originally a distribution over functions mapping $\{0,1\}^{2n}$ to $\{0,1\}$) as a distribution on $\{0,1\}^{2^{2n}}$ in the following way.

Let τ be the bijection between $[2^{2n}]$ and $\{0,1\}^{2n}$ that maps an integer $i \in [2^{2n}]$ to the i-th binary string in $\{0,1\}^{2n}$ in lexicographic order. Then a function $f : \{0,1\}^{2n} \to \{0,1\}$ is equivalent to a binary string $x^f \in \{0,1\}^{2^{2n}}$, where the i-th bit of x^f, denoted x^f_i, equals $f(\tau(i))$. Clearly this is a bijection between functions from $\{0,1\}^{2n}$ to $\{0,1\}$ and binary strings in $\{0,1\}^{2^{2n}}$.

For notational simplicity, when we say a binary string $x \in \{0,1\}^{2^{2n}}$ is drawn from $\mathcal{D}_n$, it means x is generated by first drawing a sample function $f \sim \mathcal{D}_n$ and then setting $x = x^f$.

Note that for $p \in \{0,1\}^n$, if p is the i-th binary string in $\{0,1\}^n$, then the set $B_{n,p}$ corresponds to the bits $x_{(i-1)2^n+1}, dotsc, x_{i2^n}$.

Distributional Constant-Depth Circuit Lower Bound over $\mathcal{D}_n$

Now we are ready to state our distributional circuit lower bound over $\mathcal{D}_n$ formally.

▶ **Lemma 25.** *For an integer n, let $N = 2^n$ and Sipser_d be the N-variable Sipser function as in Theorem 24.*

Consider the Boolean function $(\mathsf{Sipser}_d \circ \mathsf{OR})$ on $\{0,1\}^{N^2}$ defined as follows:
Given inputs $x_1, x_2, dotsc, x_{N^2}$, for each $1 \le i \le N$, set

$$z_i := \vee_{j=(i-1)N+1}^{iN} x_j,$$

and

$$(\mathsf{Sipser}_d \circ \mathsf{OR})(x) := \mathsf{Sipser}_d(z).$$

Then any circuit C' of depth at most $d-1$ and size at most $S = 2^{N^{\frac{1}{6(d-1)}}}$ over $\{0,1\}^{N^2}$ agrees with $(\mathsf{Sipser}_d \circ \mathsf{OR})$ with probability at most $\frac{1}{2} + N^{-\Omega(1/d)}$ when inputs are drawn from the distribution $\mathcal{D}_n$.

Before proving Lemma 25, we show that it implies the second part of Theorem 21 easily.

Proof of the second part of Theorem 21. Consider the function $(\mathsf{Sipser}_d \circ \mathsf{OR})$ defined as in Lemma 25. It is easy to see that it has a polynomial-size circuit (in fact, a formula) of depth $d+1$; and by Lemma 25, every polynomial size circuit of depth $d-1$ has at most $\frac{1}{2} + o(1)$ correlation with it when the inputs are drawn from the distribution $\mathcal{D}_n$. So it follows from the standard connection between PH and AC_0 that $\mathsf{PH}^{\mathcal{O}}$ is infinite with probability 1 when $\mathcal{O} \sim \mathcal{D}_{\mathcal{O}}$. ◀

Finally, we prove Lemma 25.

Proof of Lemma 25. By Theorem 24, there is a universal constant c, such that any circuit C of depth at most $d-1$ and size at most S over $\{0,1\}^N$ agrees with Sipser_d on at most $\left(\frac{1}{2} + N^{-c/d}\right) \cdot 2^N$ inputs.

We are going to show this lemma holds for the same c. Suppose not; then we have a circuit C of depth at most $d-1$ and size at most $S = 2^{N^{\frac{1}{6(d-1)}}}$ over $\{0,1\}^{N^2}$, such that

$$\Pr_{x\sim\mathcal{D}_n}[C(x) = (\mathsf{Sipser}_d \circ \mathsf{OR})(x)] > \frac{1}{2} + N^{-c/d}.$$

Now, for each $y_1, y_2, dotsc, y_N \in [N]^N$, we define a distribution $\mathcal{D}_n^{y_1,y_2,dotsc,y_N}$ on $\{0,1\}^{N^2}$ as follows. To generate a sample $x \sim \mathcal{D}_n^{y_1,y_2,dotsc,y_N}$, we first set $x = 0^{N^2}$. Then for each $i \in [N]$, we set $x_{(i-1)N+y_i}$ to 1 with probability $1/2$.

By construction, we can see for all x in the support of $\mathcal{D}_n^{y_1,y_2,dotsc,y_N}$,

$$(\mathsf{Sipser}_d \circ \mathsf{OR})(x) = \mathsf{Sipser}_d(x_{y_1}, x_{N+y_2}, x_{2N+y_3}, dotsc, x_{(N-1)N+y_N}).$$

Moreover, by definition, $\mathcal{D}_n$ is just the average of these distributions:

$$\mathcal{D}_n = N^{-N} \cdot \sum_{y_1,y_2,dotsc,y_N} \mathcal{D}_n^{y_1,y_2,dotsc,y_N}.$$

By an averaging argument, there exist $y_1, y_2, dotsc, y_N \in [N]^N$ such that

$$\Pr_{x\sim\mathcal{D}_n^{y_1,y_2,dotsc,y_N}}[C(x) = (\mathsf{Sipser}_d \circ \mathsf{OR})(x)] > \frac{1}{2} + N^{-c/d}.$$

Setting $x_{(i-1)N+y_i} = z_i$ for each i, and all other inputs to 0 in the circuit C, we then have a circuit D of size at most S and depth at most $d-1$ over $\{0,1\}^N$. And by the construction of $\mathcal{D}_n^{y_1,y_2,dotsc,y_N}$ and the definition of the function $(\mathsf{Sipser}_d \circ \mathsf{OR})$, we see that D agrees with Sipser_d on at least a $\frac{1}{2} + N^{-c/d}$ fraction of inputs. But this is a contradiction. ◀

6 Maximal Quantum Supremacy for Black-Box Sampling and Relation Problems

In this section we present our results about Fourier Fishing and Fourier Sampling.

We will establish an $\Omega(N/\log N)$ lower bound on the classical query complexity of Fourier Fishing, as well as an optimal $\Omega(N)$ lower bound on the classical query complexity of Fourier Sampling.

6.1 Preliminaries

We begin by introducing some useful notations. Throughout this section, given a function $f : \{0,1\}^n \to \{-1,1\}$, we define the Fourier coefficient

$$\widehat{f}(z) = 2^{-n/2} \sum_{x\in\{0,1\}^n} f(x) \cdot (-1)^{x\cdot z}$$

for each $z \in \{0,1\}^n$.

We also define

$$\mathsf{adv}(f) := 2^{-n} \cdot \sum_{z\in\{0,1\}^n, |\widehat{f}(z)|\geq 1} \widehat{f}(z)^2,$$

and set $N = 2^n$.

The following two constants will be used frequently in this section.

$$\mathsf{Succ}_Q = \frac{2}{\sqrt{2\pi}} \int_1^{+\infty} x^2 e^{-x^2/2} dx \approx 0.801 \text{ and } \mathsf{Succ}_R = \frac{2}{\sqrt{2\pi}} \int_1^{+\infty} e^{-x^2/2} dx \approx 0.317.$$

Finally, we use $\mathcal{U}_n$ to denote the uniform distribution on functions $f : \{0,1\}^n \to \{-1,1\}$.

An Approximate Formula for the Binomial Coefficients

We also need the following lemma to approximate the binomial coefficients to ease some calculations in our proofs.

▶ **Lemma 26** ((5.41) in [56]). *For value n and $|k - n/2| = o(n^{2/3})$, we have*

$$\binom{n}{k} \approx \binom{n}{n/2} \cdot e^{-\frac{(k-n/2)^2}{n/2}}$$

and

$$\ln \binom{n}{k} = \ln \binom{n}{n/2} - \frac{(k-n/2)^2}{n/2} + o(1).$$

6.2 Fourier Fishing and Fourier Sampling

We now formally define the Fourier Fishing and the Fourier Sampling problems.

▶ **Definition 27.** We are given oracle access to a function $f : \{0,1\}^n \to \{-1,1\}$.

In Fourier Sampling (or Fsampling in short), our task is to sample from a distribution $\mathcal{D}$ over $\{0,1\}^n$ such that $\|\mathcal{D} - \mathcal{D}_f\| \leq \varepsilon$, where $\mathcal{D}_f$ is the distribution defined by

$$\Pr_{\mathcal{D}_f}[y] = 2^{-n}\widehat{f}(y)^2 = \left(\frac{1}{2^n}\sum_{x\in\{0,1\}^n} f(x)(-1)^{x\cdot y}\right)^2.$$

In Fourier Fishing (or Ffishing in short), we want to find a z such that $|\widehat{f}(z)| \geq 1$. We also define a promise version of Fourier Fishing (promise-Ffishing for short), where the function f is promised to satisfy $\mathsf{adv}(f) \geq \mathsf{Succ}_Q - \frac{1}{n}$.

A Simple 1-Query Quantum Algorithm

Next we describe a simple 1-query quantum algorithm for both problems. It consists of a round of Hadamard gates, then a query to f, then another round of Hadamard gates, then a measurement in the computational basis.

The following lemma follows directly from the definitions of Fsampling and Ffishing.

▶ **Lemma 28.** *Given oracle access to a function $f : \{0,1\}^n \to \{-1,1\}$, the above algorithm solves Fsampling exactly (i.e. with $\varepsilon = 0$), and Ffishing with probability* $\mathsf{adv}(f)$.

We can now explain the meanings of the constants Succ_Q and Succ_R. When the function f is drawn from $\mathcal{U}_n$, by a simple calculation, we can see that Succ_Q is the success probability for the above simple quantum algorithm on Fourier Fishing, and Succ_R is the success probability for an algorithm outputting a uniform random string in $\{0,1\}^n$.

6.3 The $\Omega(N/\log N)$ Lower Bound for Fourier Fishing

We begin with the $\Omega(N/\log N)$ randomized lower bound for Fourier Fishing. Formally:

▶ **Theorem 29.** *There is no $o(N/\log N)$-query randomized algorithm that solves promise-Ffishing with* $\mathsf{Succ}_R + \Omega(1)$ *success probability.*

To prove Theorem 29, we first show that when the function f is drawn from $\mathcal{U}_n$, no classical algorithm with $o(N/\log N)$ queries can solve Ffishing with probability $\mathsf{Succ}_R + \Omega(1)$; we then show with high probability, a function $f \leftarrow \mathcal{U}_n$ satisfies the promise of promise-Ffishing. Formally, we have the following two lemmas.

▶ **Lemma 30.** *For large enough n,*

$$\Pr_{f \leftarrow \mathcal{U}_n}\left[\mathsf{adv}(f) < \mathsf{Succ}_Q - \frac{1}{n}\right] < \frac{1}{n}.$$

▶ **Lemma 31.** *Over $f \leftarrow \mathcal{U}_n$, no randomized algorithm with $o(N/\log N)$ queries can solve Ffishing with probability*

$$\mathsf{Succ}_R + \Omega(1).$$

Before proving these two technical lemmas, we show that they together imply Theorem 29 easily.

Proof of Theorem 29. Suppose by contradiction that there is an $o(N/\log N)$ query randomized algorithm A which has a $\mathsf{Succ}_R + \Omega(1)$ success probability for promise-Ffishing. From Lemma 30, a $1 - o(1)$ fraction of all functions from $\{0,1\}^n \to \{-1,1\}$ satisfy the promise of promise-Ffishing. Therefore, when the sample function f is drawn from $\mathcal{U}_f$, with probability $1 - o(1)$ it satisfies the promise of promise-Ffishing, and consequently A has a $\mathsf{Succ}_R + \Omega(1)$ success probability of solving Ffishing with that f. This means that A has a success probability of

$$(1 - o(1)) \cdot (\mathsf{Succ}_R + \Omega(1)) = \mathsf{Succ}_R + \Omega(1)$$

when $f \leftarrow \mathcal{U}_n$, contradicting Lemma 31. ◀

The proof of Lemma 30 is based on a tedious calculation so we defer it to Appendix C. Now we prove Lemma 31.

Proof of Lemma 31. By Yao's principle, it suffices to consider only deterministic algorithms, and we can assume the algorithm A makes exactly $t = o(N/\log N)$ queries without loss of generality.

Notations. Suppose that at the end of the algorithm, A has queried the entries in a subset $S \subseteq \{0,1\}^n$ such that $|S| = t$.

For each $z \in \{0,1\}^n$, we define

$$\widehat{f}_{\mathsf{seen}}(z) = \frac{1}{\sqrt{t}} \sum_{x \in S} f(x) \cdot (-1)^{x \cdot z}$$

and similarly

$$\widehat{f}_{\mathsf{unseen}} f(z) = \frac{1}{\sqrt{N-t}} \sum_{x \in \{0,1\}^n \setminus S} f(x) \cdot (-1)^{x \cdot z}.$$

From the definitions of $\widehat{f}(z)$, $\widehat{f}_{\mathsf{seen}}(z)$ and $\widehat{f}_{\mathsf{unseen}}(z)$, and note that $N/t = \omega(\log N) = \omega(\ln N)$, we have

$$\begin{aligned}\widehat{f}(z) &= \left(\sqrt{t} \cdot \widehat{f}_{\mathsf{seen}}(z) + \sqrt{N-t} \cdot \widehat{f}_{\mathsf{unseen}}(z)\right) \Big/ \sqrt{N} \\ &= \widehat{f}_{\mathsf{seen}}(z)/\omega(\sqrt{\ln N}) + \widehat{f}_{\mathsf{unseen}}(z) \cdot (1 - o(1)). \end{aligned} \tag{14}$$

W.h.p. $\widehat{f}_{\mathsf{seen}}(z)$ is small for all $z \in \{0,1\}^n$. We first show that, with probability at least $1 - o(1)$ over $f \leftarrow \mathcal{U}_n$, we have $|\widehat{f}_{\mathsf{seen}}(z)| \leq 2\sqrt{\ln N}$ for all $z \in \{0,1\}^n$.

Fix a $z \in \{0,1\}^n$, and note that for the algorithm A, even though which position to query next might depend on the history, the value in that position is a uniform random bit in $\{-1,1\}$. So $\widehat{f}_{\mathsf{seen}}(z)$ is a sum of t uniform i.i.d. random variables in $\{-1,1\}$.

Therefore, the probability that $|\widehat{f}_{\mathsf{seen}}(z)| > 2\sqrt{\ln N}$ for this fixed z is

$$\frac{2}{\sqrt{2\pi}} \int_{2\sqrt{\ln N}}^{+\infty} e^{-x^2/2} dx = o\left(\frac{1}{N}\right).$$

Then by a simple union bound, with probability $1 - o(1)$, there is no $z \in \{0,1\}^n$ such that $|\widehat{f}_{\mathsf{seen}}(z)| > 2\sqrt{\ln N}$ at the end of t queries. We denote the nonexistence of such a z as the event $\mathcal{E}_{\mathsf{bad}}$.

The lower bound. In the following we condition on $\mathcal{E}_{\mathsf{bad}}$. We show in this case, A cannot solve Ffishing with a success probability better than Succ_R, thereby proving the lower bound.

From (14), for each $z \in \{0,1\}^n$, we have

$$\widehat{f}(z) = o(1) + \widehat{f}_{\mathsf{unseen}}(z) \cdot (1 - o(1)).$$

Therefore, the probability of $|\widehat{f}(z)| \geq 1$ is bounded by the probability that $|\widehat{f}_{\mathsf{unseen}}(z)| \geq 1 - o(1)$. Since $\widehat{f}_{\mathsf{unseen}}(z)$ is independent of all the seen values in S, we have

$$\begin{aligned}
\Pr\left[\widehat{f}_{\mathsf{unseen}}(z) \geq 1 - o(1)\right] &= \frac{2}{\sqrt{2\pi}} \int_{1-o(1)}^{+\infty} e^{-x^2/2} dx \\
&= \frac{2}{\sqrt{2\pi}} \int_{1}^{+\infty} e^{-x^2/2} dx + o(1) \\
&= \mathsf{Succ}_R + o(1).
\end{aligned}$$

Hence, no matter which z is outputted by A, we have $|\widehat{f}(z)| \geq 1$ with probability at most $\mathsf{Succ}_R + o(1)$. That means that if we condition on $\mathcal{E}_{\mathsf{bad}}$, then A cannot solve Ffishing with probability $\mathsf{Succ}_R + \Omega(1)$. As $\mathcal{E}_{\mathsf{bad}}$ happens with probability $1 - o(1)$, this finishes the proof. ◀

6.4 The Optimal $\Omega(N)$ Lower Bound for Fourier Sampling

We first show that in fact, Lemma 31 already implies an $\Omega(N/\log N)$ lower bound for Fourier Sampling, which holds for a quite large ε.

▶ **Theorem 32.** *For any $\varepsilon < \mathsf{Succ}_Q - \mathsf{Succ}_R \approx 0.483$, the randomized query complexity for Fsampling is $\Omega(N/\log N)$.*

Proof. Note when $f \leftarrow \mathcal{U}_n$, an exact algorithm for Fsampling can be used to solve Ffishing with probability Succ_Q. Hence, a sampling algorithm for Fsampling with total variance $\leq \varepsilon$ can solve Ffishing with probability at least $\mathsf{Succ}_Q - \varepsilon$, when $f \leftarrow \mathcal{U}_n$.

Then the lower bound follows directly from Lemma 31. ◀

Next we prove the optimal $\Omega(N)$ lower bound for Fourier Sampling.

▶ **Theorem 33.** *There is a constant $\varepsilon > 0$, such that any randomized algorithm solving Fsampling with error at most ε needs $\Omega(N)$ queries.*

Proof.

Reduction to a simpler problem. Sampling problems are hard to approach, so we first reduce to a much simpler problem with Boolean output ("accept" or "reject").

Let A be a randomized algorithm for Fsampling with total variance $\leq \varepsilon$. For a function $f : \{0,1\}^n \to \{-1,1\}$ and $y \in \{0,1\}^n$, we set $p_{f,y}$ to be the probability that A outputs y with oracle access to f.

By the definition of Fsampling, for all f, we have

$$\frac{1}{2} \sum_{y \in \{0,1\}^n} \left| p_{f,y} - 2^{-n} \widehat{f}(y)^2 \right| \leq \varepsilon.$$

By an averaging argument, this implies that there exists a $y^* \in \{0,1\}^n$ such that

$$\mathop{\mathbb{E}}_{f \leftarrow \mathcal{U}_n} \left[\left| p_{f,y^*} - 2^{-n} \widehat{f}(y^*)^2 \right| \right] \leq \frac{2\varepsilon}{N}.$$

Then by Markov's inequality, we have

$$\left| p_{f,y^*} - 2^{-n} \widehat{f}(y^*)^2 \right| \leq \frac{400\varepsilon}{N},$$

for at least a 199/200 fraction of f's. Now we set $\varepsilon = \frac{1}{400} \cdot \frac{1}{100}$.

Without loss of generality, we can assume that $y^* = 0^n$. Let $z_i := \frac{1 + f(x_i)}{2}$ (where $x_1, x_2, dotsc, x_N$ is a lexicographic ordering of inputs), $Z := (z_1, dotsc, z_N)$ and $|Z| := \sum_{i=1}^{N} z_i$. Then we have

$$2^{-n} \widehat{f}(0^n)^2 = \left(\frac{2|Z|}{N} - 1 \right)^2.$$

Now we can simplify the question to one of how many z_i's the algorithm A needs to query, in order to output 0^n (we call it "accept" for convenience) with a probability $p_Z = p_{f,0^n}$ that satisfies

$$\left| p_Z - \left(\frac{2|Z|}{N} - 1 \right)^2 \right| \leq \frac{400\varepsilon}{N} \leq \frac{0.01}{N} \tag{15}$$

with probability at least 199/200 over $Z \in \{0,1\}^N$.

Analysis of the acceptance probability of A. Without loss of generality, we can assume that A non-adaptively queries t randomly-chosen inputs $z_{i_1}, z_{i_2}, dotsc, z_{i_t}$, and then accepts with a probability q_k that depends solely on $k := z_{i_1} + \cdots + z_{i_t}$. The reason is that we can change any other algorithm into this restricted form by averaging over all $N!$ permutations of Z without affecting its correctness.

Let p_w be the probability that A accepts when $|Z| = w$. Then

$$p_w = \sum_{k=0}^{t} q_k \cdot r_{k,w},$$

where $r_{k,w} := \binom{t}{k} \binom{N-t}{w-k} \Big/ \binom{N}{w}$, is the probability that $z_{i_1} + \cdots + z_{i_t} = k$ conditioned on $|Z| = w$.

Construction and Analysis of the sets U, V, W. Now, consider the following three sets:

$$U := \left\{ Z : \left| |Z| - \frac{N}{2} \right| \leq \frac{\sqrt{N}}{20} \right\},$$
$$V := \left\{ Z : \left(1 - \frac{1}{20}\right)\frac{\sqrt{N}}{2} \leq |Z| - \frac{N}{2} \leq \frac{\sqrt{N}}{2} \right\},$$
$$W := \left\{ Z : \left(1 - \frac{1}{20}\right)\sqrt{N} \leq |Z| - \frac{N}{2} \leq \sqrt{N} \right\}.$$

We calculate the probability that a uniform random Z belongs to these three sets. For a sufficiently large N, we have

$$\Pr_Z[Z \in U] \geq \mathrm{erf}\left(\frac{\sqrt{2}}{20}\right) - o(1) > 0.075,$$
$$\Pr_Z[Z \in V] \geq \frac{1}{2} \cdot \left(\mathrm{erf}\left(\frac{\sqrt{2}}{2}\right) - \mathrm{erf}\left(\frac{\sqrt{2}}{2} \cdot \frac{19}{20}\right)\right) - o(1) > 0.01,$$
$$\Pr_Z[Z \in W] \geq \frac{1}{2} \cdot \left(\mathrm{erf}(\sqrt{2}) - \mathrm{erf}\left(\sqrt{2} \cdot \frac{19}{20}\right)\right) - o(1) > 0.005.$$

Construction and Analysis of w_0, w_1, w_2. Since all $\Pr_Z[Z \in U], \Pr_Z[Z \in V], \Pr_Z[Z \in W] > 0.005$, and recall that for at least a $1 - 0.005$ fraction of Z, we have

$$\left| p_Z - \left(\frac{2|Z|}{N} - 1\right)^2 \right| \leq \frac{400\varepsilon}{N} \leq \frac{0.01}{N}.$$

So there must exist $w_0 \in U, w_1 \in V, w_2 \in W$ such that

$$\left| p_{w_i} - 4 \cdot \left(\frac{w_i - N/2}{N}\right)^2 \right| \leq \frac{0.01}{N} \tag{16}$$

for each $i \in \{0, 1, 2\}$.

To ease our calculation, let $u_i = \dfrac{w_i - N/2}{\sqrt{N}}$, then we have $w_i = N/2 + u_i\sqrt{N}$. By the definition of the u_i's, we also have $|u_0| \leq \dfrac{1}{20}, u_1 \in [0.475, 0.5], u_2 \in [0.95, 1]$.

Plugging in u_i's, for each $i \in \{0, 1, 2\}$, equation (16) simplifies to

$$\left| p_{w_i} - \frac{4u_i^2}{N} \right| \leq \frac{0.01}{N}. \tag{17}$$

We can calculate the ranges of the p_{w_i}'s by plugging the ranges of the u_i's,

$$p_{w_0} \leq \frac{0.02}{N},$$
$$p_{w_1} \in \left[\frac{0.95^2 - 0.01}{N}, frac1 + 0.01N\right] \subseteq \left[\frac{0.89}{N}, frac1.01N\right],$$
$$p_{w_2} \in \left[\frac{4 \cdot 0.95^2 - 0.01}{N}, frac4 + 0.01N\right] \subseteq \left[\frac{3.6}{N}, frac4.01N\right].$$

We are going to show that the above is impossible when $t = o(N)$. That is, one cannot set the q_k's in such a way that all p_{w_i}'s satisfy the above constraints when $t = o(N)$.

It is safe to set q_k to zero when $|k-t/2|$ is large. To simplify the matters, we first show that we can set nearly all the q_k's to zero. By the Chernoff bound without replacement, for each w_i and large enough c we have

$$\begin{aligned}
&\sum_{k:|k-t/2|>c\sqrt{t}} r_{k,w_i} \\
&= \Pr\Big[\big|z_{i_1}+\cdots+z_{i_t}-\tfrac{t}{2}\big| \geq c\sqrt{t} : |Z| = w_i = \tfrac{N}{2}+u_i\sqrt{N}\Big] \\
&\leq \Pr\Big[\Big|z_{i_1}+\cdots+z_{i_t}-\Big(\tfrac{t}{2}+\tfrac{u_i t}{\sqrt{N}}\Big)\Big| \geq c\sqrt{t}-\Big|\Big(\tfrac{t}{2}+\tfrac{u_i t}{\sqrt{N}}\Big)-\tfrac{t}{2}\Big| : |Z| = w_i = \tfrac{N}{2}+u_i\sqrt{N}\Big] \\
&\leq \Pr\Big[\Big|z_{i_1}+\cdots+z_{i_t}-\Big(\tfrac{t}{2}+\tfrac{u_i t}{\sqrt{N}}\Big)\Big| \geq c\sqrt{t}-|u_i|\sqrt{t} : |Z| = \tfrac{N}{2}+u_i\sqrt{N}\Big] \qquad (\tfrac{t}{\sqrt{N}} \leq \sqrt{t}) \\
&\leq \exp\Big\{-2\tfrac{(c\sqrt{t}-|u_i|\sqrt{t})^2}{t}\Big\} \\
&= \exp\big\{-2(c-|u_i|)^2\big\} \\
&\leq \exp\big\{\Omega(c^2)\big\}.
\end{aligned}$$

Then we can set $c = c_1\sqrt{\ln N}$ for a sufficiently large constant c_1, so that for all w_i's,

$$\sum_{k:|k-t/2|>c\sqrt{t}} r_{k,w_i} \leq \frac{1}{N^2}.$$

This means that we can simply set all q_k's with $|k-t/2| > c\sqrt{t} = c_1\sqrt{t\ln N}$ to zero, and only consider k such that $|k-t/2| \leq c_1\sqrt{t\ln N}$, as this only changes each p_{w_i} by a negligible value. From now on, we call an integer k *valid*, if $|k-t/2| \leq c_1\sqrt{t\ln N}$.

Either $\frac{r_{k,w_0}}{r_{k,w_1}} \geq 0.05$ or $\frac{r_{k,w_2}}{r_{k,w_1}} \geq 10$. Now, we are going to show the most technical part of this proof: for all valid k, we have either

$$\frac{r_{k,w_0}}{r_{k,w_1}} \geq 0.05 \text{ or } \frac{r_{k,w_2}}{r_{k,w_1}} \geq 10. \tag{18}$$

Suppose for contradiction that there is a valid k that satisfies

$$\frac{r_{k,w_0}}{r_{k,w_1}} < 0.05 \text{ and } \frac{r_{k,w_2}}{r_{k,w_1}} < 10. \tag{19}$$

Estimation of r_{k,w_i}'s. We first use Lemma 26 to derive an accurate estimate of $\ln r_{k,w_i}$ for each w_i.

We set $N_t = N - t$ for simplicity. Recall that

$$r_{k,w} = \binom{t}{k}\binom{N_t}{w-k}\Big/\binom{N}{w}.$$

For each w_i, since $|k-t/2| \leq c_1\sqrt{t\ln N}$ and $t = o(N)$, we have

$$\left|w_i - k - \frac{N_t}{2}\right| \leq \left|w_i - \frac{N}{2}\right| + \left|k - \frac{t}{2}\right| \leq u_i\sqrt{N} + c_1\sqrt{t\ln N} = o(N_t^{2/3}),$$

and note that $|w_i - N/2| = |u_i\sqrt{N}| = o(N^{2/3})$. So we can apply Lemma 26 to derive

$$\begin{aligned}
\ln r_{k,w_i} &= \ln\binom{t}{k} + \ln\binom{N_t}{w_i-k} - \ln\binom{N}{w_i} \\
&= -\frac{(w_i-k-N_t/2)^2}{N_t/2} + \frac{(w_i-N/2)^2}{N/2} + C + \ln\binom{t}{k} + o(1),
\end{aligned}$$

in which C is a constant that does not depend on k or w_i.

Let $d = (k - t/2)/\sqrt{t}$ (so $k = t/2 + d\sqrt{t}$), and recall that $w_i = N/2 + u_i\sqrt{N}$ for each w_i. We can further simplify the expression as

$$\begin{aligned}\ln r_{k,w_i} &= -\tfrac{(N/2+u_i\sqrt{N}-t/2-d\sqrt{t}-N_t/2)^2}{N_t/2} + \tfrac{(N/2+u_i\sqrt{N}-N/2)^2}{N/2} + C + \ln\binom{t}{k} + o(1)\\ &= -\tfrac{(u_i\sqrt{N}-d\sqrt{t})^2}{N_t/2} + 2u_i^2 + C + \ln\binom{t}{k} + o(1).\end{aligned}$$

Estimation of $\frac{r_{k,w_j}}{r_{k,w_i}}$. Note that $N_t = N - t = (1 - o(1))N$. So we can approximate the ratio between two r_{k,w_i} and r_{k,w_j} by

$$\begin{aligned}\ln\frac{r_{k,w_j}}{r_{k,w_i}} &= \ln r_{k,w_j} - \ln r_{k,w_i}\\ &= -\frac{(u_j\sqrt{N} - d\sqrt{t})^2}{N_t/2} + 2u_j^2 + \frac{(u_i\sqrt{N} - d\sqrt{t})^2}{N_t/2} - 2u_i^2 + o(1)\\ &= 2u_j^2 - 2u_i^2 + \frac{((u_i+u_j)\sqrt{N} - 2d\sqrt{t})(u_i-u_j)\sqrt{N}}{N_t/2} + o(1)\\ &= 2u_j^2 - 2u_i^2 + 2(u_i^2 - u_j^2) - 4d\frac{\sqrt{tN}}{N_t}(u_i - u_j) + o(1)\\ &= -4d\frac{\sqrt{tN}}{N_t}(u_i - u_j) + o(1).\end{aligned}$$

Verifying (18). Finally, to simplify matters further, we set $x = -4d\frac{\sqrt{tN}}{N_t}$, and substitute it in (19) for k. We have

$$\ln\frac{r_{k,w_0}}{r_{k,w_1}} = x(u_1 - u_0) + o(1) < -\ln 20,$$

which simplifies to

$$x < \frac{-\ln 20}{u_1 - u_0} + o(1) \le \frac{-\ln 20}{0.505} + o(1) \le -5.93 + o(1).$$

Similarly, we have

$$\ln\frac{r_{k,w_2}}{r_{k,w_1}} = x(u_1 - u_2) + o(1) < \ln 10$$

and

$$x > -\frac{\ln 10}{u_2 - u_1} - o(1) \ge -\frac{\ln 10}{0.45} - o(1) \ge -5.12 - o(1).$$

contradiction.

The lower bound. So (18) holds for all valid k, which means for all k such that $|k - t/2| \le c_1\sqrt{t\ln N}$, either $\frac{r_{k,w_0}}{r_{k,w_1}} \ge 0.05$ or $\frac{r_{k,w_2}}{r_{k,w_1}} \ge 10$.

Let H be the set of all valid integers k. We set

$$S = \left\{k \in H : \frac{r_{k,w_0}}{r_{k,w_1}} \ge 0.05\right\} \text{ and } T = H \setminus S.$$

By (18), for any $k \in T$, we have $\frac{r_{k,w_2}}{r_{k,w_1}} \geq 10$.

Since $p_{w_1} = \sum_{k \in S} q_k \cdot r_{k,w_1} + \sum_{k \in T} q_k \cdot r_{k,w_1} \geq \frac{0.89}{N}$ (recall we have set all q_k's to zero for $k \notin H$), we must have either $\sum_{k \in S} q_k \cdot r_{k,w_1} \geq \frac{0.445}{N}$ or $\sum_{k \in T} q_k \cdot r_{k,w_1} \geq \frac{0.445}{N}$.

If $\sum_{k \in S} q_k \cdot r_{k,w_1} \geq \frac{0.445}{N}$, we have

$$p_{w_0} \geq \sum_{k \in S} q_k \cdot r_{k,w_1} \cdot \frac{r_{k,w_0}}{r_{k,w_1}} \geq \frac{0.445}{N} \cdot 0.05 \geq \frac{0.022}{N},$$

which contradicts the constraint that $p_{w_0} \leq \frac{0.02}{N}$. Otherwise, $\sum_{k \in T} q_k \cdot r_{k,w_1} \geq \frac{0.445}{N}$; then

$$p_{w_2} \geq \sum_{k \in T} q_k \cdot r_{k,w_1} \cdot \frac{r_{k,w_2}}{r_{k,w_1}} \geq \frac{0.445}{N} \cdot 10 \geq \frac{4.45}{N},$$

which violates the requirement that $p_{w_2} \leq \frac{4.01}{N}$.

Since both cases lead to a contradiction, A needs to make $\Omega(N)$ queries and this completes the proof. ◀

7 Quantum Supremacy Relative to Efficiently-Computable Oracles

We now discuss our results about quantum supremacy relative to oracles in P/poly.

Building on work by Zhandry [61] and Servedio and Gortler [53], we first show that, if (classical) one-way functions exist, then there exists an oracle $\mathcal{O} \in \mathsf{P/poly}$ such that $\mathsf{BPP}^{\mathcal{O}} \neq \mathsf{BQP}^{\mathcal{O}}$. Then we make a connection to the previous section by showing that, assuming the existence of (classical) subexponentially strong one-way functions, Fourier Fishing and Fourier Sampling are hard even when it is promised that the oracle is in P/poly.

We also study several other complexity questions relative to P/poly oracles: for example, P vs NP, P vs BPP, and BQP vs SZK. Since these questions are not connected directly with quantum supremacy, we will discuss them in Appendix A.

7.1 Preliminaries

Recall that an oracle $\mathcal{O} : \{0,1\}^* \to \{0,1\}$ is itself a language, so we say that an oracle $\mathcal{O}$ is in P/poly when the corresponding language belongs to P/poly, and we use $\mathcal{O}_n$ to denote its restriction to $\{0,1\}^n$.

Given two sets $\mathcal{X}$ and $\mathcal{Y}$, we define $\mathcal{Y}^{\mathcal{X}}$ as the set of functions $f : \mathcal{X} \to \mathcal{Y}$. For a set $\mathcal{X}$, we will sometimes abuse notation and write $\mathcal{X}$ to denote the uniform distribution on $\mathcal{X}$.

(Quantum) Pseudorandom Functions and Permutations

We are going to use pseudorandom functions and permutations throughout this section, so we first review their definitions.

▶ **Definition 34** (PRF and PRP)**.** A pesudorandom function is a function $\mathsf{PRF} : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$, where $\mathcal{K}$ is the key-space, and $\mathcal{X}$ and $\mathcal{Y}$ are the domain and the range. $\mathcal{K}, domain, image$ are implicitly functions of the security parameter n.[10] We write $y = \mathsf{PRF}_k(x)$.

[10] We denote them by $\mathcal{K}_n, domain_n, image_n$ when we need to be clear about the security parameter n.

Similarly, a pesudorandom permutation is a function $\mathsf{PRP} : \mathcal{K} \times \mathcal{X} \to \mathcal{X}$, where $\mathcal{K}$ is the key-space, and $\mathcal{X}$ is the domain of the permutation. $\mathcal{K}$ and $\mathcal{X}$ are implicitly functions of the security parameter n. We write $y = \mathsf{PRP}_k(x)$. It is guaranteed that PRP_k is a permutation on $\mathcal{X}$ for each $k \in \mathcal{K}$.

For simplicity, we use $\mathsf{PRF}_{\mathcal{K}}$ to denote the distribution on functions $f : \mathcal{X} \to \mathcal{Y}$ by drawing $k \leftarrow \mathcal{K}$ and set $f := \mathsf{PRF}_k$.

We now introduce the definitions of classical and quantum security.

▶ **Definition 35** (Classical-Security)**.** A pseudorandom function $\mathsf{PRF} : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ is (classically) secure if no classical adversary A can distinguish between a truly random function and the function PRF_k for a random k in polynomial time. That is, for every such A, there exists a negligible function $\varepsilon = \varepsilon(n)$ such that

$$\left| \Pr_{k \leftarrow \mathcal{K}}[A^{\mathsf{PRF}_k}() = 1] - \Pr_{f \leftarrow \mathcal{Y}^{\mathcal{X}}}[A^{f}() = 1] \right| < \varepsilon.$$

Also, we say that a pseudorandom function PRF is exponentially-secure, if the above holds even for classical adversaries that take $2^{O(n)}$ time.

Similarly, a pseudorandom permutation PRP is (classically) secure if no classical adversary A can distinguish between a truly random permutation and the function PRP_k for a random k in polynomial time.

Sometimes, especially in the context of one-way functions, we will talk about *subexponential* security. By this we simply mean that there is no adversary running in $2^{n^{o(1)}}$ time.

▶ **Definition 36** (Quantum-Security)**.** A pseudorandom function PRF is quantum-secure if no quantum adversary A making quantum queries can distinguish between a truly random function and the function PRF_k for a random k in polynomial time.

Also, a pseudorandom permutation PRP is quantum-secure if no quantum adversary A making quantum queries can distinguish between a truly random permutation and the function PRP_k for a random k in polynomial time.

On the Existence of PRFs

It is well-known that the existence of one-way functions implies the existence of PRFs and PRPs.

▶ **Lemma 37** ([34, 32, 33, 42])**.** *If one-way functions exist, then there exist secure PRFs and PRPs. Similarly, if subexponentially-secure one-way functions exist, then there exist exponentially-secure PRFs.*

We remark here that these are all *purely classical* assumptions, which make no reference to quantum algorithms. Also, the latter assumption is the same one as in the famous *natural proofs* barrier [48].

7.2 A Construction from Zhandry [61]

To prove our separations, we will use a construction from Zhandry [61] with some modifications. We first construct a PRP and a PRF, and summarize some of their useful properties.

Definitions of $\mathsf{PRP}^{\mathsf{raw}}$ and $\mathsf{PRF}^{\mathsf{mod}}$

Assuming one-way functions exist, by Lemma 37, let $\mathsf{PRP}^{\mathsf{raw}}$ be a secure pesudorandom permutation with key-space $\mathcal{K}^{\mathsf{raw}}$ and domain $\mathcal{X}^{\mathsf{raw}}$. We interpret $\mathcal{X}^{\mathsf{raw}}$ as $[N]$, where $N = N(n) = |\mathcal{X}^{\mathsf{raw}}|$.

Then we define another pseudorandom function $\mathsf{PRF}^{\mathsf{mod}}_{(k,a)}(x) = \mathsf{PRP}^{\mathsf{raw}}_k((x-1) \bmod a + 1)$ where:

- The key space of $\mathsf{PRF}^{\mathsf{mod}}$ is $\mathcal{K}^{\mathsf{mod}} = \mathcal{K}^{\mathsf{raw}} \times \mathcal{A}$ where $\mathcal{A}$ is the set of primes in $[\sqrt{N}/4, sqrtN/2]$.
- The domain and image are both $\mathcal{X}^{\mathsf{raw}}$, that is, $\mathcal{X}^{\mathsf{mod}} = \mathcal{X}^{\mathsf{raw}}$ and $\mathcal{Y}^{\mathsf{mod}} = \mathcal{X}^{\mathsf{raw}}$.

Note that we denote the latter one by $\mathsf{PRF}^{\mathsf{mod}}$ (not $\mathsf{PRP}^{\mathsf{mod}}$) because it is no longer a PRP.

Properties of $\mathsf{PRP}^{\mathsf{raw}}$ and $\mathsf{PRF}^{\mathsf{mod}}$

We now summarize several properties of $\mathsf{PRP}^{\mathsf{raw}}$ and $\mathsf{PRF}^{\mathsf{mod}}$, which can be proved along the same lines as [61].

▶ **Lemma 38** (Implicit in Claim 1 and Claim 2 of [61]). *The following statements hold when $\mathsf{PRP}^{\mathsf{raw}}$ is classical secure.*

1. *Both $\mathsf{PRP}^{\mathsf{raw}}$ and $\mathsf{PRF}^{\mathsf{mod}}$ are classical secure PRFs. Consequently, no classical algorithm A can distinguish them with a non-negligible advantage.*
2. *Given oracle access to $\mathsf{PRF}^{\mathsf{mod}}_{(k,a)}$ where $(k,a) \leftarrow \mathcal{K}^{\mathsf{mod}}$, there is a quantum algorithm that can recover a with probability at least $1-\varepsilon$.*
3. *There is a quantum algorithm that can distinguish $\mathsf{PRP}^{\mathsf{raw}}$ from $\mathsf{PRF}^{\mathsf{mod}}$ with advantage $1-\varepsilon$.*

Here $\varepsilon = \varepsilon(n)$ is a negligible function.

For completeness, we prove Lemma 38 in Appendix D, by adapting the proofs of Claims 1 and 2 in [61].

7.3 BPP vs. BQP

Next we discuss whether there is an oracle $\mathcal{O} \in \mathsf{P/poly}$ that separates BPP from BQP. We show that the answer is yes provided that one-way functions exist.

▶ **Theorem 39.** *Assuming one-way functions exist, there exists an oracle $\mathcal{O} \in P/poly$ such that $BPP^{\mathcal{O}} \neq BQP^{\mathcal{O}}$.*

Proof. We are going to use $\mathsf{PRP}^{\mathsf{raw}}$ and $\mathsf{PRF}^{\mathsf{mod}}$ from Section 7.2.

The oracle $\mathcal{O}$ will encode the truth tables of functions $f_1, f_2, dotsc$, where each f_n is a function from $\mathcal{X}^{\mathsf{raw}}_n$ to $\mathcal{X}^{\mathsf{raw}}_n$. For each n, with probability 0.5 we draw f_n from $\mathsf{PRP}^{\mathsf{raw}}_{\mathcal{K}^{\mathsf{raw}}}$, that is, draw $k \leftarrow \mathcal{K}^{\mathsf{raw}}$ and set $f_n := \mathsf{PRP}^{\mathsf{raw}}_k$, and with probability 0.5 we draw f_n from $\mathsf{PRF}^{\mathsf{mod}}_{\mathcal{K}^{\mathsf{mod}}}$ similarly. We set L to be the unary language consisting of all 0^n for which f_n is drawn from $\mathsf{PRP}^{\mathsf{raw}}_{\mathcal{K}^{\mathsf{raw}}}$.

By Lemma 38, there exists a BQP machine $M^{\mathcal{O}}$ that decides L correctly on all but finite many values of n with probability 1. Since we can simply hardwire the values of n on which $M^{\mathcal{O}}$ is incorrect, it follows that $L \in \mathsf{BQP}^{\mathcal{O}}$ with probability 1.

On the other hand, again by Lemma 38, no BPP machine can distinguish $\mathsf{PRP}^{\mathsf{raw}}_{\mathcal{K}^{\mathsf{raw}}}$ and $\mathsf{PRF}^{\mathsf{mod}}_{\mathcal{K}^{\mathsf{mod}}}$ with a non-negligible advantage. So let M be a BPP machine, and let $E_n(M)$ be the event that M decides whether $0^n \in L$ correctly. We have

$$\Pr_{\mathcal{O}}[E_n(M)] = \frac{1}{2} + o(1),$$

even conditioning on events $E_1(M), dotsc, E_{n-1}(M)$. Therefore, we have $\Pr_{\mathcal{O}}[\wedge_{i=1}^{+\infty} E_n(M)] = 0$, which means that a BPP machine M decides L with probability 0. Since there are countably many BPP machines, it follows that $L \notin \mathsf{BPP}^{\mathcal{O}}$ with probability 1. Hence $\mathsf{BPP}^{\mathcal{O}} \neq \mathsf{BQP}^{\mathcal{O}}$ with probability 1.

Finally, note that each f_n has a polynomial-size circuit, and consequently $\mathcal{O} \in \mathsf{P/poly}$. ◀

7.4 Fourier Fishing and Fourier Sampling

Finally, we discuss Fourier Fishing and Fourier Sampling. We are going to show that, assuming the existence of subexponentially-secure one-way functions, Fourier Fishing and Fourier Sampling are hard even when it is promised that the oracle belongs to P/poly.

▶ **Theorem 40.** *Assuming the existence of subexponentially strong one-way functions, there is no polynomial-time classical algorithm that can solve promise-Ffishing with probability*

$$\mathsf{Succ}_R + \Omega(1),$$

even when it is promised that the oracle function belongs to P/poly.

Proof. By Lemma 37, we can use our one-way function to construct an exponentially-secure pseudorandom function, $\mathsf{PRF} : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$. Without loss of generality, we assume that $|\mathcal{Y}| = 2$ and $|\mathcal{X}| = 2^n$. Then we interpret $\mathcal{X}$ as the set $\{0,1\}^n$, and $\mathcal{Y}$ as the set $\{-1,1\}$.

A Concentration Inequality. Now, consider the distribution $\mathsf{PRF}_{\mathcal{K}}$ on functions $\{0,1\}^n \to \{-1,1\}$. We claim that

$$\Pr_{f \leftarrow \mathsf{PRF}_{\mathcal{K}}}[\mathsf{adv}(f) > \mathsf{Succ}_Q - 1/n] > 1 - \frac{1}{n} - o(1). \tag{20}$$

To see this: from Lemma 30, we have

$$\Pr_{f \leftarrow \mathcal{Y}^{\mathcal{X}}}[\mathsf{adv}(f) > \mathsf{Succ}_Q - 1/n] > 1 - \frac{1}{n}.$$

Therefore, if (20) does not hold, then we can construct a distinguisher between $\mathsf{PRF}_{\mathcal{K}}$ and truly random functions $\mathcal{X}^{\mathcal{Y}}$ by calculating $\mathsf{adv}(f)$ in $2^{O(n)}$ time. But this contradicts the assumption that PRF is exponentially-secure.

A distributional lower bound. Next, we show that for every polynomial-time algorithm A, we have

$$\Pr_{f \leftarrow \mathsf{PRF}_{\mathcal{K}}}[A^f \text{ solves Ffishing correctly}] \leq \mathsf{Succ}_R + o(1). \tag{21}$$

This is because when f is a truly random function, from Lemma 31, we have

$$\Pr_{f \leftarrow \mathcal{Y}^{\mathcal{X}}}[A^f \text{ solve Ffishing correctly}] \leq \mathsf{Succ}_R + o(1).$$

So if (21) does not hold, then we can construct a distinguisher between $\mathsf{PRF}_{\mathcal{K}}$ and truly random functions $\mathcal{X}^{\mathcal{Y}}$ by simulating A^f to get its output z, and then checking whether z is a correct solution to Ffishing in $2^{O(n)}$ time. This again contradicts our assumption that PRF is exponentially-secure.

The lower bound. Finally, we prove the theorem. Suppose for contradiction that there is such a polynomial-time algorithm A. Then when $f \leftarrow \mathsf{PRF}_{\mathcal{K}}$, from (20), with probability $1 - 1/n - o(1)$, we have that f satisfies the promise of promise-Ffishing. Thus, A solves Ffishing when $f \leftarrow \mathsf{PRF}_{\mathcal{K}}$ with probability at least

$$(1 - o(1)) \cdot (\mathsf{Succ}_R + \Omega(1)) = \mathsf{Succ}_R + \Omega(1),$$

which contradicts (21). ◀

By a similar reduction, we can show that Fourier Sampling is also hard.

▶ **Corollary 41.** *Assuming the existence of subexponentially-secure one-way functions, no polynomial-time classical algorithm can solve Fsampling with error*

$$\varepsilon < \mathsf{Succ}_Q - \mathsf{Succ}_R \approx 0.483,$$

even if it is promised that the oracle function belongs to P/poly.

Proof. For a function f, an exact algorithm for Fsampling can be used to solve Ffishing with probability $\mathsf{adv}(f)$. Hence, a polynomial-time sampling algorithm A for Fsampling with error at most ε can solve Ffishing with probability at least $\mathsf{adv}(f) - \varepsilon$.

Note that by (20), when $f \leftarrow \mathsf{PRF}_{\mathcal{K}}$, the algorithm A can solve Ffishing with probability at least

$$(\mathsf{Succ}_Q - \frac{1}{n} - \varepsilon) \cdot (1 - o(1)) = \mathsf{Succ}_Q - o(1) - \varepsilon.$$

Therefore, by (21), we must have $\varepsilon \geq \mathsf{Succ}_Q - \mathsf{Succ}_R$, which completes the proof. ◀

8 Complexity Assumptions Are Needed for Quantum Supremacy Relative to Efficiently-Computable Oracles

In Section 7.4, we showed that the existence of subexponentially-secure one-way functions implies that Fourier Sampling and Fourier Fishing are classically hard, even when it is promised that the oracle function belongs to P/poly. We also showed that if one-way functions exist, then there exists an oracle $\mathcal{O} \in \mathsf{P/poly}$ which separates BPP from BQP.

It is therefore natural to ask whether we can prove the same statements *unconditionally*. In this section, we show that at least *some* complexity assumptions are needed.

▶ **Theorem 42.** *Suppose* $\mathsf{SampBPP} = \mathsf{SampBQP}$ *and* $\mathsf{NP} \subseteq \mathsf{BPP}$. *Then for every oracle* $\mathcal{O} \in \mathsf{P/poly}$, *we have* $\mathsf{SampBPP}^{\mathcal{O}} = \mathsf{SampBQP}^{\mathcal{O}}$ *(and consequently* $\mathsf{BPP}^{\mathcal{O}} = \mathsf{BQP}^{\mathcal{O}}$*).*

Much like in the proof of Theorem 21, we need to show that under the stated assumptions, every SampBQP algorithm M can be simulated by a SampBPP algorithm A.

▶ **Lemma 43.** *Suppose* $\mathsf{SampBPP} = \mathsf{SampBQP}$ *and* $\mathsf{NP} \subseteq \mathsf{BPP}$. *Then for any polynomial* $q(n)$ *and any* SampBQP *oracle algorithm* M, *there is a* SampBPP *oracle algorithm* A *such that:*

For every $\mathcal{O} \in \mathsf{SIZE}(q(n))$,[11] *let* $\mathcal{D}^M_{x,varepsilon}$ *and* $\mathcal{D}^A_{x,varepsilon}$ *be the distributions output by* $M^{\mathcal{O}}$ *and* $A^{\mathcal{O}}$ *respectively on input* $\langle x, 0^{1/\varepsilon} \rangle$. *Then*

$$\|\mathcal{D}^M_{x,varepsilon} - \mathcal{D}^A_{x,varepsilon}\| \leq \varepsilon.$$

[11] A language is in $\mathsf{SIZE}(q(n))$ if it can be computed by circuits of size $q(n)$.

Before proving Lemma 43, we show that it implies Theorem 42.

Proof of Theorem 42. Let $\mathcal{O} \in \mathsf{P/poly}$ be an oracle. Then there exists a polynomial $q(n)$ such that $\mathcal{O} \in \mathsf{SIZE}(q(n))$.

Let $\mathcal{S}$ be a sampling problem in $\mathsf{SampBQP}^{\mathcal{O}}$. This means that there is a $\mathsf{SampBQP}$ oracle algorithm M, such that for all $x \in \{0,1\}^*$ and ε, we have $\|\mathcal{D}^M_{x,varepsilon} - \mathcal{S}_x\| \leq \varepsilon$. Let A_M be the corresponding $\mathsf{SampBPP}$ algorithm whose existence we've assumed, and consider the following algorithm A': given input $\langle x, 0^{1/\varepsilon}\rangle$, run A_M on input $\langle x, 0^{2/\varepsilon}\rangle$ to get a sample from $\mathcal{D}^{A_M}_{x,varepsilon/2}$.

Then we have

$$\begin{aligned}\|\mathcal{D}^{A'}_{x,varepsilon} - \mathcal{S}_x\| &= \|\mathcal{D}^{A_M}_{x,varepsilon/2} - \mathcal{S}_x\| \\ &\leq \|\mathcal{D}^{M}_{x,varepsilon/2} - \mathcal{D}^{A_M}_{x,varepsilon/2}\| + \|\mathcal{D}^{M}_{x,varepsilon/2} - \mathcal{S}_x\| \\ &\leq 2 \cdot \frac{\varepsilon}{2} \leq \varepsilon.\end{aligned}$$

This means that A' solves $\mathcal{S}$ and $\mathcal{S} \in \mathsf{SampBPP}^{\mathcal{O}}$. Hence $\mathsf{SampBQP}^{\mathcal{O}} \subseteq \mathsf{SampBPP}^{\mathcal{O}}$. ◀

Now we prove Lemma 43. The simulation procedure is similar to that in Lemma 23: that is, we replace each oracle gate, one by one, by a known function while minimizing the introduced error. The difference is that, instead of the brute-force method as in Lemma 23, here we use a more sophisticated PAC learning subroutine to find an "approximator" to replace the oracle gates.

Proof of Lemma 43. Let $\mathcal{O} \in \mathsf{SIZE}(q(n))$; we let $f_n = \mathcal{O}_n$ for simplicity.

Recall that there exists a fixed polynomial p, such that given input $\langle x, 0^{1/\varepsilon}\rangle$, the machine M first constructs a quantum circuit C with $N = p(|x|, 1/\varepsilon)$ qubits and N gates classically (C can contain $\mathcal{O}$ gates). Without loss of generality, we can assume for each n, all f_n gates act only on the first n qubits.

For a function $f : \{0,1\}^k \to \{0,1\}$, recall that U_f denotes the unitary operator mapping $|i\rangle$ to $(-1)^{f(i)}|i\rangle$ for $i \in \{0,1\}^k$.

Suppose there are T $\mathcal{O}$-gates in total, and the i-th $\mathcal{O}$-gate is an f_{n_i} gate. Then the unitary operator U applied by the circuit C can be decomposed as

$$U = U_{T+1}(U_{f_{n_T}} \otimes I_{N-n_T}) \cdots (U_{f_{n_2}} \otimes I_{N-n_2}) U_2 (U_{f_{n_1}} \otimes I_{N-n_1}) U_1,$$

where the U_i's are the unitary operators corresponding to the sub-circuits which don't contain an $\mathcal{O}$-gate.

Again, the algorithm proceeds by replacing each $\mathcal{O}$-gate by a much simpler gate one by one, without affecting the resulting quantum state too much, and then simulating the final circuit to get a sample to output.

Replacing the t-th $\mathcal{O}$-gate. Suppose we have already replaced the first $t-1$ $\mathcal{O}$-gates: that is, for each $i \in [t-1]$, we replaced the f_{n_i} gate (the i-th $\mathcal{O}$-gate) with a g_i gate. Now we are going to replace the t-th $\mathcal{O}$-gate.

Let

$$|v\rangle = U_t(U_{g_{t-1}} \otimes I_{N-n_{t-1}}) \cdots (U_{g_2} \otimes I_{N-n_2}) U_2 (U_{g_1} \otimes I_{N-n_1}) U_1 |0\rangle^{\otimes N},$$

which is the quantum state right before the t-th $\mathcal{O}$ gate in the circuit after the replacement.

For brevity, we use f to denote the function f_{n_t}, and we drop the subscript t of n_t when it is clear from context.

Analysis of incurred error. The t-th $\mathcal{O}$-gate is an f gate. If we replace it by a g gate, then the deviation caused to the quantum states is

$$\|U_f \otimes I_{N-n}|v\rangle - U_g \otimes I_{N-n}|v\rangle\| = \|(U_f - U_g) \otimes I_{N-n}|v\rangle\|.$$

Let H be the Hilbert space corresponding to the last $N-n$ qubits, and let $\rho = \mathrm{Tr}_H[|v\rangle\langle v|]$. Then proceeding exactly as in Lemma 23, we have

$$\|((U_f - U_g) \otimes I_{N-n})|v\rangle\|^2 = 4 \cdot \Pr_{i\sim Q}[f(i) \neq g(i)], \tag{22}$$

where Q is the probability on $\{0,1\}^n$ defined by $Q(i) = \langle i|\rho|i\rangle$, and $[f(i) \neq g(i)]$ is the indicator function that takes value 1 when $f(i) \neq g(i)$ and 0 otherwise.

Upper bounding the deviation (22) vis PAC learning. Now, we want to replace f by another function g, so that the deviation term (22) is minimized.

By a standard result of PAC learning (cf. the book of Vapnik [59]), for parameters ε_1 and δ_1, we can take a $\mathrm{poly}(n, varepsilon_1^{-1}, ln\delta_1^{-1})$ number of i.i.d. samples from Q, and then find a function g in $\mathsf{SIZE}(q(n))$ which agrees with f on those samples. Then with probability at least $1-\delta_1$, we will have

$$\Pr_{i\sim Q}[f(i) \neq g(i)] \leq \varepsilon_1.$$

The choice of ε_1 and δ_1 will be made later. In any case, with probability at least $1-\delta_1$, we have

$$\|(U_f - U_g) \otimes I_{N-n}|v\rangle\|^2 \leq 4\varepsilon_1,$$

which in turn implies

$$\|(U_f - U_g) \otimes I_{N-n}|v\rangle\| \leq 2 \cdot \sqrt{\varepsilon_1}.$$

Analysis of the final circuit C^{final}. Suppose that at the end, for each $t \in [T]$, our algorithm has replaced the t-th $\mathcal{O}$-gate with a g_t gate, where g_t is a function from $\{0,1\}^{n_t}$ to $\{0,1\}$. Let C^{final} be the circuit after the replacement. Also, let

$$V = U_{T+1}(U_{g_T} \otimes I_{N-n_T}) \cdots (U_{g_2} \otimes I_{N-n_2})U_2(U_{g_1} \otimes I_{N-n_1})U_1$$

be the unitary operator corresponding to C^{final}.

Now we set $\delta_1 = \dfrac{\varepsilon}{2T}$, and $\varepsilon_1 = \dfrac{\varepsilon^4}{256T^2}$. Then by a union bound over all rounds, and following exactly the same analysis as in Lemma 23, with probability at least $1-T\cdot\delta_1 = 1-\varepsilon/2$, we have

$$\|U|0\rangle^{\otimes N} - V|0\rangle^{\otimes N}\| \leq 2T \cdot \sqrt{\varepsilon_1} = \frac{\varepsilon^2}{8}.$$

Our classical algorithm A then simulates stages 2 and 3 of the $\mathsf{SampBQP}$ algorithm M straightforwardly. It first takes a sample z by measuring $V|0\rangle^{\otimes N}$ in the computational basis, and then outputs $A^{\mathsf{output}}(z)$ as its sample, where A^{output} is the classical algorithm used by M in stage 3.

By Corollary 5, with probability at least $1-\varepsilon/2$, the final distribution $\mathcal{D}$ on which A takes samples satisfies

$$\|\mathcal{D} - \mathcal{D}^M_{x,varepsilon}\| \leq \sqrt{2 \cdot \frac{\varepsilon^2}{8}} = \frac{\varepsilon}{2}.$$

Hence, the outputted distribution $\mathcal{D}^A_{x,varepsilon}$ satisfies

$$\|\mathcal{D}^A_{x,varepsilon} - \mathcal{D}^M_{x,varepsilon}\| \leq \varepsilon.$$

Showing that A is a SampBPP algorithm. We still have to show that A is a SampBPP oracle algorithm. From the previous discussion, A needs to do the following non-trivial computations.

- Taking a polynomial number of samples from Q. This task is in SampBQP (no oracle involved) by definition. By our assumption SampBQP = SampBPP, it can be done in SampBPP.
- Finding a $g \in \mathsf{SIZE}(q(n))$ such that g agrees with f on all the samples. This can be done in NP, so by our assumption NP $\subseteq$ BPP, it can be done in BPP.
- Taking a sample by measuring $V|0\rangle^{\otimes N}$. Again, this task is in SampBQP, and hence can be done in SampBPP by our assumption.

Therefore, A is a SampBPP oracle algorithm. ◀

9 Open Problems

There are many exciting open problems left by this paper; here we mention just a few.

1. Is QUATH (our assumption about the hardness of guessing whether $|\langle 0|C|0\rangle|^2$ is greater or less than the median) true or false?
2. Is Conjecture 1 true? That is, does a random quantum circuit on n qubits sample an unbalanced distribution over n-bit strings with $1 - 1/\exp(n)$ probability?
3. We showed that there exists an oracle relative to which SampBPP = SampBQP but PH is infinite. Can we nevertheless show that SampBPP = SampBQP would collapse PH in the unrelativized world? (An affirmative answer would, of course, follow from Aaronson and Arkhipov's Permanent-of-Gaussians Conjecture [3], as mentioned in Section 1.2.)
4. Is our classical algorithm to simulate a quantum circuit with n qubits and m gates optimal? Or could we reduce the complexity, say from $m^{O(n)}$ to $2^{O(n)} \cdot m^{O(1)}$, while keeping the space usage polynomial? Does it matter if we only want to sample from the output distribution, rather than actually calculating the probabilities? What about if we only want to guess an amplitude with small bias, as would be needed to refute QUATH?
5. For random quantum circuit sampling, we proved a conditional hardness result that talks directly about the observed outputs of a sampling process, rather than about the unknown distribution that's sampled from. Can we get analogous hardness results for the BosonSampling or IQP models, under some plausible hardness conjecture? Note that the argument from Section 3 doesn't work directly for BosonSampling or IQP, for the simple reason that in those models, the advantage over chance in guessing a given amplitude is at least $1/\exp(n)$, rather than $1/\exp(m)$ for some $m \gg n$ as is the case for random circuits.
6. We proved a lower bound of $\Omega(N)$ on the classical query complexity of Fourier Sampling, for a rather small error $\varepsilon = \frac{1}{40000}$. The error constant does matter for sampling problems, since there is no efficient way to reduce the error in general. So can we discover the *exact threshold* ε for an $\Omega(N)$ lower bound? That is, find the constant ε such that there is an $o(N)$ query classical algorithm solving Fourier Sampling with error ε, but any classical algorithm with error $< \varepsilon$ needs $\Omega(N)$ queries?
7. In Section 7, we showed that there is an oracle $\mathcal{O}$ in P/poly separating BPP from BQP, assuming that one-way functions exist. Is it possible to weaken the assumption to, say, NP $\not\subset$ BPP?

Acknowledgments We thank Shalev Ben-David, Sergio Boixo, Yuzhou Gu, Greg Kuperberg, John Martinis, Ashley Montanaro, John Preskill, Vadim Smelyansky, Ronald de Wolf, and Mark Zhandry for helpful discussions about the subject of this paper.

References

1 S. Aaronson. BQP and the polynomial hierarchy. In *Proc. ACM STOC*, 2010. arXiv:0910.4698.
2 S. Aaronson. Google, D-wave, and the case of the factor-10^8 speedup for WHAT?, 2015. URL: `http://www.scottaaronson.com/blog/?p=2555`.
3 S. Aaronson and A. Arkhipov. The computational complexity of linear optics. *Theory of Computing*, 9(4):143–252, 2013. Earlier version in Proc. ACM STOC'2011. ECCC TR10-170, arXiv:1011.3245.
4 S. Aaronson et al. The Complexity Zoo. URL: `http://www.complexityzoo.com`.
5 S. Aaronson and Y. Shi. Quantum lower bounds for the collision and the element distinctness problems. *J. of the ACM*, 51(4):595–605, 2004.
6 S. Aaronson and A. Wigderson. Algebrization: a new barrier in complexity theory. *ACM Trans. on Computation Theory*, 1(1), 2009. Earlier version in Proc. ACM STOC'2008.
7 Sccot Aaronson. New evidence that quantum mechanics is hard to simulate on classical computers. `http://www.scottaaronson.com/talks/newev.ppt`, 2009.
8 Scott Aaronson. The equivalence of sampling and searching. *Theory of Computing Systems*, 55(2):281–298, 2014.
9 Scott Aaronson and Andris Ambainis. Forrelation: A problem that optimally separates quantum from classical computing. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*, pages 307–316. ACM, 2015.
10 Scott Aaronson, Shalev Ben-David, and Robin Kothari. Separations in query complexity using cheat sheets. *arXiv preprint arXiv:1511.01937*, 2015.
11 Scott Aaronson, Adam Bouland, Greg Kuperberg, and Saeed Mehraban. The computational complexity of ball permutations. *arXiv preprint arXiv:1610.06646*, 2016.
12 D. Aharonov and M. Ben-Or. Fault-tolerant quantum computation with constant error. In *Proc. ACM STOC*, pages 176–188, 1997. quant-ph/9906129.
13 D. Aharonov, M. Ben-Or, and E. Eban. Interactive proofs for quantum computations. arXiv:0810.5375, 2008.
14 Andris Ambainis. Polynomial degree and lower bounds in quantum complexity: Collision and element distinctness with small range. *Theory of Computing*, 1(1):37–46, 2005.
15 Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009.
16 Theodore Baker, John Gill, and Robert Solovay. Relativizations of the P=?NP question. *SIAM Journal on computing*, 4(4):431–442, 1975.
17 C. Bennett, E. Bernstein, G. Brassard, and U. Vazirani. Strengths and weaknesses of quantum computing. *SIAM J. Comput.*, 26(5):1510–1523, 1997. quant-ph/9701001.
18 E. Bernstein and U. Vazirani. Quantum complexity theory. *SIAM J. Comput.*, 26(5):1411–1473, 1997. Earlier version in Proc. ACM STOC'1993.
19 Sergio Boixo, Sergei V Isakov, Vadim N Smelyanskiy, Ryan Babbush, Nan Ding, Zhang Jiang, John M Martinis, and Hartmut Neven. Characterizing quantum supremacy in near-term devices. *arXiv preprint arXiv:1608.00263*, 2016.
20 Dan Boneh and Richard J. Lipton. Quantum cryptanalysis of hidden linear functions. In *Annual International Cryptology Conference*, pages 424–437. Springer, 1995.
21 Fernando G. S. L. Brandão, Aram W. Harrow, and Michał Horodecki. Local random quantum circuits are approximate polynomial-designs. *Communications in Mathematical Physics*, 346(2):397–434, 2016.

22 Sergey Bravyi and David Gosset. Improved classical simulation of quantum circuits dominated by clifford gates. *arXiv preprint arXiv:1601.07601*, 2016.

23 M. Bremner, R. Jozsa, and D. Shepherd. Classical simulation of commuting quantum computations implies collapse of the polynomial hierarchy. *Proc. Roy. Soc. London*, A467(2126):459–472, 2010. arXiv:1005.1407.

24 Michael J. Bremner, Ashley Montanaro, and Dan J. Shepherd. Average-case complexity versus approximate simulation of commuting quantum computations. *arXiv preprint arXiv:1504.07999*, 2015.

25 Michael J. Bremner, Ashley Montanaro, and Dan J. Shepherd. Achieving quantum supremacy with sparse and noisy commuting quantum computations. *arXiv preprint arXiv:1610.01808*, 2016.

26 A. Broadbent, J. Fitzsimons, and E. Kashefi. Universal blind quantum computation. In *Proc. IEEE FOCS*, 2009. arXiv:0807.4154.

27 Jacques Carolan, Christopher Harrold, Chris Sparrow, Enrique Martín-López, Nicholas J Russell, Joshua W Silverstone, Peter J Shadbolt, Nobuyuki Matsuda, Manabu Oguma, Mikitaka Itoh, Graham D Marshall, Mark G Thompson, Jonathan C F Matthews, Toshikazu Hashimoto, Jeremy L O'Brien, and Anthony Laing. Universal linear optics. *Science*, 349(6249):711–716, 2015.

28 Lijie Chen. A note on oracle separations for BQP. *arXiv preprint arXiv:1605.00619*, 2016.

29 Edward Farhi and Aram W. Harrow. Quantum supremacy through the quantum approximate optimization algorithm. *arXiv preprint arXiv:1602.07674*, 2016.

30 L. Fortnow and J. Rogers. Complexity limitations on quantum computation. *J. Comput. Sys. Sci.*, 59(2):240–252, 1999. cs.CC/9811023.

31 Keisuke Fujii. Noise threshold of quantum supremacy. *arXiv preprint arXiv:1610.03632*, 2016.

32 O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *J. of the ACM*, 33(4):792–807, 1986. Earlier version in Proc. IEEE FOCS'1984, pp. 464-479.

33 Oded Goldreich and Leonid A Levin. A hard-core predicate for all one-way functions. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 25–32. ACM, 1989.

34 J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

35 Johan Hastad. Almost optimal lower bounds for small depth circuits. In *Proceedings of the eighteenth annual ACM symposium on Theory of computing*, pages 6–20. ACM, 1986.

36 R. Impagliazzo and A. Wigderson. P=BPP unless E has subexponential circuits: derandomizing the XOR Lemma. In *Proc. ACM STOC*, pages 220–229, 1997.

37 Richard Jozsa and Marrten Van den Nest. Classical simulation complexity of extended clifford circuits. *Quantum Information & Computation*, 14(7&8):633–648, 2014.

38 Gil Kalai. How quantum computers fail: quantum codes, correlations in physical systems, and noise accumulation. *arXiv preprint arXiv:1106.0485*, 2011.

39 J. Kelly, R. Barends, A. G. Fowler, A. Megrant, E. Jeffrey, T. C. White, D. Sank, J. Y. Mutus, B. Campbell, Yu Chen, et al. State preservation by repetitive error detection in a superconducting quantum circuit. *Nature*, 519(7541):66–69, 2015.

40 Samuel Kutin. Quantum lower bound for the collision problem with small range. *Theory of Computing*, 1(1):29–36, 2005.

41 Leonid A. Levin. The tale of one-way functions. *Problems of Information Transmission*, 39(1):92–103, 2003.

42 Michael Luby and Charles Rackoff. How to construct pseudorandom permutations from pseudorandom functions. *SIAM Journal on Computing*, 17(2):373–386, 1988.

43 Igor L. Markov and Yaoyun Shi. Simulating quantum computation by contracting tensor networks. *SIAM Journal on Computing*, 38(3):963–981, 2008.

44 Tomoyuki Morimae, Keisuke Fujii, and Joseph F Fitzsimons. Hardness of classically simulating the one-clean-qubit model. *Physical review letters*, 112(13):130502, 2014.

45 Noam Nisan and Avi Wigderson. Hardness vs randomness. *Journal of computer and System Sciences*, 49(2):149–167, 1994.

46 Borja Peropadre, Gian Giacomo Guerreschi, Joonsuk Huh, and Alán Aspuru-Guzik. Microwave boson sampling. *arXiv preprint arXiv:1510.08064*, 2015.

47 John Preskill. Quantum computing and the entanglement frontier. *arXiv preprint arXiv:1203.5813*, 2012.

48 A. A. Razborov and S. Rudich. Natural proofs. *J. Comput. Sys. Sci.*, 55(1):24–35, 1997. Earlier version in Proc. ACM STOC'1994, pp. 204-213.

49 Benjamin Rossman, Rocco A Servedio, and Li-Yang Tan. An average-case depth hierarchy theorem for boolean circuits. In *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on*, pages 1030–1048. IEEE, 2015.

50 Terry Rudolph. Why I am optimistic about the silicon-photonic route to quantum computing. *arXiv preprint arXiv:1607.08535*, 2016.

51 Amit Sahai and Salil Vadhan. A complete problem for statistical zero knowledge. *Journal of the ACM (JACM)*, 50(2):196–249, 2003.

52 W. J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *J. Comput. Sys. Sci.*, 4(2):177–192, 1970.

53 Rocco A Servedio and Steven J Gortler. Equivalences and separations between quantum and classical learnability. *SIAM Journal on Computing*, 33(5):1067–1092, 2004.

54 A. Shamir. IP=PSPACE. *J. of the ACM*, 39(4):869–877, 1992. Earlier version in Proc. IEEE FOCS'1990, pp. 11-15.

55 P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, 1997. Earlier version in Proc. IEEE FOCS'1994. quant-ph/9508027.

56 Joel Spencer. *Asymptopia*, volume 71. American Mathematical Soc., 2014.

57 B. M. Terhal and D. P. DiVincenzo. Adaptive quantum computation, constant-depth circuits and Arthur-Merlin games. *Quantum Information and Computation*, 4(2):134–145, 2004. quant-ph/0205133.

58 S. Toda. PP is as hard as the polynomial-time hierarchy. *SIAM J. Comput.*, 20(5):865–877, 1991. Earlier version in Proc. IEEE FOCS'1989, pp. 514-519.

59 Vladimir Naumovich Vapnik. *Statistical learning theory*, volume 1. Wiley New York, 1998.

60 Andrew Chi-Chih Yao. Separating the polynomial-time hierarchy by oracles. In *26th Annual Symposium on Foundations of Computer Science (sfcs 1985)*, 1985.

61 Mark Zhandry. How to construct quantum random functions. In *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*, pages 679–687. IEEE, 2012.

62 Mark Zhandry. A note on quantum-secure prps. *arXiv preprint arXiv:1611.05564*, 2016.

A Other Results on Oracle Separations in P/poly

In this section we discuss the rest of our results on complexity theory relative to oracles in $\mathsf{P/poly}$ (see Figure 1 for an overview). For the definitions of the involved complexity classes, see for example [4].

We first discuss P and NP. We observe that there exists an oracle $\mathcal{O} \in \mathsf{P/poly}$ such that $\mathsf{P}^{\mathcal{O}} \neq \mathsf{NP}^{\mathcal{O}}$ *unconditionally*, and no oracle $\mathcal{O} \in \mathsf{P/poly}$ can make $\mathsf{P} = \mathsf{NP}$ unless $\mathsf{NP} \subset \mathsf{P/poly}$.

Then we discuss P and BPP. We first prove that the standard derandomization assumption (there exists a function $f \in \mathsf{E} = \mathsf{DTIME}(2^{O(n)})$ that requires a $2^{\Omega(n)}$-size circuit) also implies

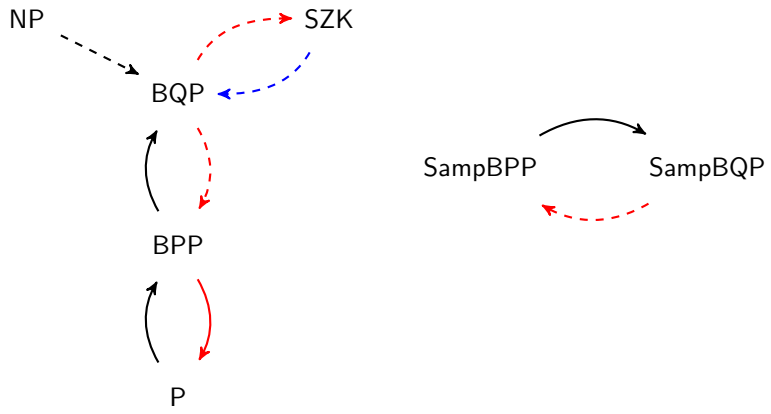


Figure 1 $\mathcal{C}_1 \to \mathcal{C}_2$ indicates $\mathcal{C}_1$ is contained in $\mathcal{C}_2$ respect to every oracle in P/poly, and $\mathcal{C}_1 \dashrightarrow \mathcal{C}_2$ denotes that there is an oracle $\mathcal{O} \in \mathsf{P/poly}$ such that $\mathcal{C}_1^{\mathcal{O}} \not\subset \mathcal{C}_2^{\mathcal{O}}$. Red indicates this statement is based on the existence of *classical one-way functions*, Blue indicates the statement is based on the existence of *quantum one-way functions*, and Black indicates the statement holds unconditionally.

that $\mathsf{P}^{\mathcal{O}} = \mathsf{BPP}^{\mathcal{O}}$ for all $\mathcal{O} \in \mathsf{P/poly}$. Then, surprisingly, we show that the converse also holds! I.e., if no such f exists, then there exists an oracle $\mathcal{O} \in \mathsf{P/poly}$ such that $\mathsf{P}^{\mathcal{O}} \neq \mathsf{BPP}^{\mathcal{O}}$.

Finally, we discuss BQP and SZK. We show that assuming the existence of one-way functions, there exist oracles in P/poly that separate BQP from SZK, and also SZK from BQP.

We will need to use quantum-secure pseudorandom permutations. By a very recent result of Zhandry [62], their existence follows from the existence of quantum one-way functions.

▶ **Lemma 44** ([62]). *Assuming quantum one way functions exist, there exist quantum-secure PRPs.*

A.1 P, BPP, BQP vs. NP

We begin with the relationships of P, BPP, and BQP to NP relative to oracles in P/poly.

The first observation is that using the function OR and standard diagonalization techniques, together with the fact that OR is hard for quantum algorithms [17], we immediately have:

▶ **Observation 45.** *There is an oracle $\mathcal{O} \in P/poly$ such that $NP^{\mathcal{O}} \not\subset BQP^{\mathcal{O}}$.*

On the other side, we also show that unless $\mathsf{NP} \subset \mathsf{P/poly}$ ($\mathsf{BQP/poly}$), there is no oracle $\mathcal{O} \in \mathsf{P/poly}$ such that $\mathsf{NP}^{\mathcal{O}} \subseteq \mathsf{BPP}^{\mathcal{O}}$ ($\mathsf{BQP}^{\mathcal{O}}$).

▶ **Theorem 46.** *Unless $NP \subset P/poly$, there is no oracle $\mathcal{O} \in P/poly$ such that $NP^{\mathcal{O}} \subseteq BPP^{\mathcal{O}}$. Likewise, there is no oracle $\mathcal{O} \in P/poly$ such that $NP^{\mathcal{O}} \subseteq BQP^{\mathcal{O}}$ unless $NP \subseteq BQP/poly$.*

Proof. Suppose there is an oracle $\mathcal{O} \in \mathsf{P/poly}$ such that $\mathsf{NP}^{\mathcal{O}} \subseteq \mathsf{BPP}^{\mathcal{O}}$. Since $\mathsf{BPP} \subset \mathsf{P/poly}$, and $\mathsf{P}^{\mathcal{O}}/\mathsf{poly} \subseteq \mathsf{P/poly}$ (since the relevant parts of the oracle $\mathcal{O}$ can be directly supplied to the P/poly algorithm), we have $\mathsf{NP} \subseteq \mathsf{NP}^{\mathcal{O}} \subset \mathsf{P/poly}$. The second claim can be proved in the same way. ◀

The following corollary is immediate.

▶ **Corollary 47.** *There is an oracle $\mathcal{O} \in P/poly$ such that $P^{\mathcal{O}} \neq NP^{\mathcal{O}}$, and there is no oracle $\mathcal{O} \in P/poly$ such that $P^{\mathcal{O}} = NP^{\mathcal{O}}$ unless $NP \subset P/poly$.*

A.2 P vs. BPP

Next we consider the relationship between P and BPP. It is not hard to observe that the standard derandomization assumption for P = BPP is in fact strong enough to make $\mathsf{P}^{\mathcal{O}} = \mathsf{BPP}^{\mathcal{O}}$ for every oracle $\mathcal{O}$ in P/poly.

Given a function $f : \{0,1\}^n \to \{0,1\}$, let $H_{\text{wrs}}(f)$ be the minimum size of circuits computing f exactly.

▶ **Observation 48** (Implicit in [45, 36], see also Theorem 20.7 in [15])**.** *If there exists a function $f \in E = DTIME(2^{O(n)})$ and $\varepsilon > 0$ such that $H_{wrs}(f) \geq 2^{\varepsilon n}$ for sufficiently large n, then $BPP^{\mathcal{O}} = P^{\mathcal{O}}$ for every $\mathcal{O} \in P/poly$.*

Proof Sketch. From [45] and [36], the assumption leads to a strong PRG which is able to fool circuits of a fixed polynomial size with a logarithmic seed length.

An algorithm with an oracle $\mathcal{O} \in \mathsf{P/poly}$ with a certain input can still be represented by a polynomial size circuit, so we can still enumerate all possible seeds to get a deterministic algorithm. ◀

Surprisingly, we show that condition is not only sufficient, but also necessary.

▶ **Theorem 49.** *If for every $f \in E = DTIME(2^{O(n)})$ and $\varepsilon > 0$, there are infinitely many n's with $H_{wrs}(f) < 2^{\varepsilon n}$, then there exists an oracle $\mathcal{O} \in P/poly$ such that $BPP^{\mathcal{O}} \neq P^{\mathcal{O}}$.*

Proof. For simplicity, in the following we will specify an oracle $\mathcal{O}$ by a sequence of functions $\{f_i\}$, where each f_i is a function from $\{0,1\}^{n_i} \to \{0,1\}$ and the sequence $\{n_i\}$ is strictly increasing. That is, $\mathcal{O}_{n_i}$ is set to f_i, and $\mathcal{O}$ maps all strings with length not in $\{n_i\}$ to 0.

As there are only countably many P oracle TM machines, we let $\{A_i\}_{i=1}^{+\infty}$ be an ordering of them.

The GapMaj function. Recall that the gapped-majority function, $\mathsf{GapMaj} : \{0,1\}^N \to \{0,1\}$, which outputs 1 if the input has Hamming weight $\geq 2N/3$, or 0 if the input has Hamming weight $\leq N/3$, and is undefined otherwise, is the function which separates P and BPP in the query complexity world. We are going to encode inputs to GapMaj in the oracle bits to achieve our separation.

We call an oracle *valid*, if for each n, either $|\mathcal{O}_n^{-1}(0)| \geq \frac{2}{3} \cdot 2^n$ or $|\mathcal{O}_n^{-1}(1)| \geq \frac{2}{3} \cdot 2^n$. That is, if we interpret $\mathcal{O}_n$ as a binary string with length 2^n, then $\mathsf{GapMaj}(\mathcal{O}_n)$ is defined.

The language $L^{\mathcal{O}}$. For a valid oracle $\mathcal{O}$, we define the following language:

$$L_{\mathcal{O}} = \{0^n : \mathsf{GapMaj}(\mathcal{O}_n) = 1\}.$$

Clearly, this language lies in $\mathsf{BPP}^{\mathcal{O}}$. To prove the theorem, we will construct a valid oracle $\mathcal{O}$ such that $L_{\mathcal{O}} \notin \mathsf{P}^{\mathcal{O}}$.

Construction of $\mathcal{O}$. To construct such an oracle, we resort to the standard diagonalization method: for each integer i, we find an integer n_i and set the function $\mathcal{O}_{n_i}$ so that the machine A_i can't decide 0^{n_i} correctly. In order to do this, we will make sure that each A_i can only see 0 when querying the function $\mathcal{O}_{n_i}$. Since A_i can only see a polynomial number of bits, we can set the remaining bits in $\mathcal{O}_{n_i}$ adversarially.

Let $\mathcal{O}^i_{\mathsf{part}}$ be the oracle specified by $\{\mathcal{O}_{n_j}\}_{j=1}^i$, and let T_i be the maximum integer such that a bit in $\mathcal{O}_{T_i}$ is queried by A_i when running on input 0^{n_i}. Observe that by setting $n_{i+1} > T_i$, we can make sure that $A_i^{\mathcal{O}}(0^{n_i}) = A_i^{\mathcal{O}^i_{\mathsf{part}}}(0^{n_i})$ for each i.

Diagonalization against A_i. Suppose we have already constructed $\mathcal{O}_{n_1}, dotsc, oracle_{n_{i-1}}$, and we are going to deal with A_i. Since A_i is a P machine, there exists a constant c such that A_i runs in at most n^c steps for inputs with length n. Thus, A_i can query at most n^c values in $\mathcal{O}_n$ on input 0^n.

Construction and Analysis of f. Now consider the following function f, which analyzes the behavior of $A_i^{\mathcal{O}_{\mathsf{part}}^{i-1}}$:

- given an input $x \in \{0,1\}^*$, let $m = |x|$;
- the first $m_1 = \lfloor m/5c \rfloor$ bits of x encode an integer $n \in [2^{m_1}]$;
- the next $m_2 = m - m_1$ bits of x encode a string $p \in \{0,1\}^{m_2}$;
- $f(x) = 1$ iff $A_i^{\mathcal{O}_{\mathsf{part}}^{i-1}}(0^n)$ has queried $\mathcal{O}_n(z)$ for an $z \in \{0,1\}^n$ with p as a prefix.[12]

It is not hard to see that $f \in \mathsf{E}$: the straightforward algorithm which directly simulates $A_i^{\mathcal{O}_{\mathsf{part}}^{i-1}}(0^n)$ runs in $O(n^c) = 2^{O(m/5c \cdot c)} = 2^{O(m)}$ time (note that the input length is $m = |x|$). Therefore, by our assumption, there exists an integer m such that $2^{\lfloor m/5c \rfloor} > \max(T_{i-1}, n_{i-1})$ and $H_{\mathrm{wrs}}(f_m) < 2^{m/c}$. Then we set $n_i = 2^{\lfloor m/5c \rfloor}$.

Construction and Analysis of $\mathcal{O}_{n_i}$. Now, if $A_i^{\mathcal{O}_{\mathsf{part}}^{i-1}}(0^{n_i}) = 1$, we set $\mathcal{O}_{n_i}$ to be the constant function $\mathbf{0}$, so that $L^{\mathcal{O}}(0^{n_i}) = 0$.

Otherwise, $A_i^{\mathcal{O}_{\mathsf{part}}^{i-1}}(0^{n_i}) = 0$. We define a function $g : \{0,1\}^{n_i} \to \{0,1\}$ as follows: $g(z) = 1$ iff $A_i^{\mathcal{O}_{\mathsf{part}}^{i-1}}(0^{n_i})$ has queried $\mathcal{O}_{n_i}(z')$ for an $z' \in \{0,1\}^{n_i}$ such that z and z' share a prefix of length $m - \lfloor m/5c \rfloor$. Note that $g(z)$ can be implemented by hardwiring n_i and $z_{1\ldots m-\lfloor m/5c \rfloor}$ (that is, the first $m - \lfloor m/5c \rfloor$ bits of z) into the circuit for f_m, which means that there is a circuit of size $2^{m/c} = n_i^{O(1)}$ for g. We set $\mathcal{O}_{n_i} := \neg g$.

From the definition of g and the fact that $A_i^{\mathcal{O}_{\mathsf{part}}^{i-1}}(0^{n_i})$ makes at most n_i^c queries, there is at most a

$$\frac{n_i^c}{2^{m-\lfloor m/5c \rfloor}} < \frac{n_i^c}{2^{4c\lfloor m/5c \rfloor}} = n_i^{-3c}$$

fraction of inputs that are 0 in $\neg g$. Hence, $\mathsf{GapMaj}(\neg g) = 1$ and $L^{\mathcal{O}}(0^{n_i}) = 1$.

We claim that in both cases, we have $A_i^{\mathcal{O}_{\mathsf{part}}^{i-1}}(0^{n_i}) = A_i^{\mathcal{O}_{\mathsf{part}}^{i}}(0^{n_i})$. This holds trivially in the first case since we set $\mathcal{O}_{n_i} := \mathbf{0}$. For the second case, note from the definition of g that all queries by $A_i^{\mathcal{O}_{\mathsf{part}}^{i}}(0^{n_i})$ to $\mathcal{O}_{n_i}$ return 0, and hence A_i will behave exactly the same.

Finally, since we set $n_i > T_{i-1}$ for each i, we have $A_i^{\mathcal{O}}(0^{n_i}) = A_i^{\mathcal{O}_{\mathsf{part}}^{i}}(0^{n_i}) \neq L^{\mathcal{O}}(0^{n_i})$, which means that no A_i can decide $L^{\mathcal{O}}$. ◀

A.3 BQP vs. SZK

Next we investigate the relationship between BQP and SZK relative to oracles in $\mathsf{P/poly}$. We first show that, by using quantumly-secure pseudorandom permutations, as well as the quantum lower bound for distinguishing permutations from 2-to-1 functions [5], we can construct an oracle in $\mathsf{P/poly}$ which separates SZK from BQP.

▶ **Theorem 50.** *Assuming quantum-secure one way functions exist, there exists an oracle $\mathcal{O} \in P/poly$ such that $SZK^{\mathcal{O}} \not\subset BQP^{\mathcal{O}}$.*

[12] For simplicity, we still use $\mathcal{O}_n$ to denote the restriction of $\mathcal{O}_{\mathsf{part}}^{i-1}$ on $\{0,1\}^n$.

Proof. Let PRP be a quantum-secure pseudorandom permutation from $\mathcal{K} \times \mathcal{X} \to \mathcal{X}$, whose existence is guaranteed by Lemma 44.

We first build a pseudorandom 2-to-1 function from PRP. We interpret $\mathcal{X}$ as $[N]$ where $N = |\mathcal{X}|$, and assume that N is even. We construct $\mathsf{PRF}^{2\to1} : (\mathcal{K} \times \mathcal{K}) \times \mathcal{X} \to \mathcal{X}$ as follows:

- The key space $\mathcal{K}^{2\to1}$ is $\mathcal{K} \times \mathcal{K}$. That is, a key $k \in \mathcal{K}^{2\to1}$ is a pair of keys (k_1, k_2).
- $\mathsf{PRF}^{2\to1}_{(k_1,k_2)}(x) := \mathsf{PRP}_{k_2}((\mathsf{PRP}_{k_1}(x) \bmod N/2) + 1)$.

Note that $\mathsf{PRF}^{2\to1}$ would be a uniformly random 2-to-1 function from $[N] \to [N]$, if PRP_{k_1} and PRP_{k_2} were replaced by two uniformly random permutations on $[N]$. Hence, by a standard reduction argument, $\mathsf{PRF}^{2\to1}$ is a quantumly-secure pseudorandom 2-to-1 function. That is, for any polynomial-time quantum algorithm A, we have

$$\left| \Pr_{k\leftarrow\mathcal{K}^{2\to1}}[A^{\mathsf{PRF}^{2\to1}_k}() = 1] - \Pr_{f\leftarrow\mathsf{F}^{2\to1}_{\mathcal{X}}}[A^f() = 1] \right| < \varepsilon,$$

where ε is a negligible function and $\mathsf{F}^{2\to1}_{\mathcal{X}}$ is the set of 2-to-1 functions from $\mathcal{X} \to \mathcal{X}$.

Also, from the definition of PRP, we have

$$\left| \Pr_{k\leftarrow\mathcal{K}^{\mathsf{PRP}}}[A^{\mathsf{PRP}_k}() = 1] - \Pr_{f\leftarrow\mathsf{Perm}_{\mathcal{X}}}[A^f() = 1] \right| < \varepsilon,$$

where $\mathsf{Perm}_{\mathcal{X}}$ is the set of permutations on $\mathcal{X}$.

From the results of Aaronson and Shi [5], Ambainis [14] and Kutin [40], no $o(N^{1/3})$-query quantum algorithm can distinguish a random permutation from a random 2-to-1 function. Therefore, we have

$$\left| \Pr_{f\leftarrow\mathsf{F}^{2\to1}_{\mathcal{X}}}[A^f() = 1] - \Pr_{f\leftarrow\mathsf{Perm}_{\mathcal{X}}}[A^f() = 1] \right| < o(1).$$

Putting the above three inequalities together, we have

$$\left| \Pr_{k\leftarrow\mathcal{K}^{2\to1}}[A^{\mathsf{PRF}^{2\to1}_k}() = 1] - \Pr_{k\leftarrow\mathcal{K}^{\mathsf{PRP}}}[A^{\mathsf{PRP}_k}() = 1] \right| < o(1),$$

which means A cannot distinguish $\mathsf{PRF}^{2\to1}_{\mathcal{K}^{2\to1}}$ and $\mathsf{PRP}_{\mathcal{K}^{\mathsf{PRP}}}$.

On the other side, an SZK algorithm can easily distinguish a permutation from a two-to-one function. Therefore, we can proceed exactly as in Theorem 39 to construct an oracle $\mathcal{O} \in \mathsf{P/poly}$ such that $\mathsf{SZK}^{\mathcal{O}} \not\subset \mathsf{BQP}^{\mathcal{O}}$. ◀

Very recently, Chen [28] showed that, based on a construction similar to the "cheat-sheet" function by Aaronson, Ben-David and Kothari [10], we can take any function which is hard for BPP algorithms, and turn it into a function which is hard for SZK algorithms in a black-box fashion. We are going to adapt this construction, together with a PRF, to build an oracle in $\mathsf{P/poly}$ which separates BQP from SZK.

▶ **Theorem 51.** *Assuming one-way functions exist, there exists an oracle $\mathcal{O} \in P/poly$ such that $BQP^{\mathcal{O}} \not\subset SZK^{\mathcal{O}}$.*

Proof. We will use the $\mathsf{PRF}^{\mathsf{mod}} : \mathcal{K}^{\mathsf{mod}} \times \mathcal{X}^{\mathsf{mod}} \to \mathcal{X}^{\mathsf{mod}}$ defined in Section 7.2 here. For simplicity, we will use $\mathcal{X}$ to denote $\mathcal{X}^{\mathsf{mod}}$ in this proof. Recall that $\mathcal{X}$ is interpreted as $[N]$ for $N = N(n) = |\mathcal{X}|$.

Construction of distributions $\mathcal{D}_n^i$. For each n, we define distributions $\mathcal{D}_n^0$ and $\mathcal{D}_n^1$ on $(\mathcal{X}_n \to \mathcal{X}_n) \times \{0,1\}^{\sqrt{N}/2}$ as follows. We draw a function $f_n : \mathcal{X} \to \mathcal{X}$ from $\mathsf{PRF}^{\mathsf{mod}}_{\mathcal{K}^{\mathsf{mod}}}$, that is, we draw $(k,a) \leftarrow \mathcal{K}^{\mathsf{mod}} = \mathcal{K}^{\mathsf{raw}} \times A$, and set $f_n := \mathsf{PRF}^{\mathsf{mod}}_{(k,a)}$; then we let $z = 0^{\sqrt{N}/2}$ first, and set $z_a = i$ in $\mathcal{D}_n^i$; finally we output the pair (f,z) as a sample.

Distinguishing $\mathcal{D}_n^0$ and $\mathcal{D}_n^1$ is hard for SZK. Recall that SZK is a semantic class. That is, a given protocol Π might be invalid with different oracles or different inputs (i.e., the protocol might not satisfy the zero-knowledge constraint, or the verifier might accept with a probability that is neither $\geq 2/3$ nor $\leq 1/3$). We write $\Pi^{(f,z)}() = \perp$ when Π is invalid given oracle access to (f,z).

We claim that for any protocol Π, one of the following two claims must hold for sufficiently large n:

(A) $\Pr_{(f,z)\leftarrow\mathcal{D}_n^0}\left[\Pi^{(f,z)}() = \perp\right] > 0.1$ or $\Pr_{(f,z)\leftarrow\mathcal{D}_n^1}\left[\Pi^{(f,z)}() = \perp\right] > 0.1$.

(B) $\left|\Pr_{(f,z)\leftarrow\mathcal{D}_n^0}\left[\Pi^{(f,z)}() = 1\right] - \Pr_{(f,z)\leftarrow\mathcal{D}_n^1}\left[\Pi^{(f,z)}() = 1\right]\right| < 0.2$.

That is, either Π is invalid on a large fraction of oracles, or else Π cannot distinguish $\mathcal{D}_n^0$ from $\mathcal{D}_n^1$ with a very good probability.

Building a BPP algorithm to break $\mathsf{PRF}^{\mathsf{mod}}$. Suppose for a contradiction that there are infinitely many n such that none of (A) and (B) hold. Without loss of generality, we can assume that

$$\Pr_{(f,z)\leftarrow\mathcal{D}_n^1}\left[\Pi^{(f,z)}() = 1\right] - \Pr_{(f,z)\leftarrow\mathcal{D}_n^0}\left[\Pi^{(f,z)}() = 1\right] \geq 0.2.$$

We are going to build a BPP algorithm which is able to break $\mathsf{PRF}^{\mathsf{mod}}$ on those n, thereby contradicting Lemma 38.

From (A), we have

$$\Pr_{(f,z)\leftarrow\mathcal{D}_n^1}\left[\Pi^{(f,z)}() = 1\right] - \left(1 - \Pr_{(f,z)\leftarrow\mathcal{D}_n^0}\left[\Pi^{(f,z)}() = 0\right]\right) \geq 0.1,$$

which simplifies to

$$\Pr_{(f,z)\leftarrow\mathcal{D}_n^1}\left[\Pi^{(f,z)}() = 1\right] + \Pr_{(f,z)\leftarrow\mathcal{D}_n^0}\left[\Pi^{(f,z)}() = 0\right] \geq 1.1.$$

From the definition of $\mathcal{D}_n^0$ and $\mathcal{D}_n^1$, the above implies that

$$\Pr_{(k,a)\leftarrow\mathcal{K}^{\mathsf{mod}}}\left[\Pi^{(f,z_1)}() = 1 \text{ and } \Pi^{(f,z_0)}() = 0, f = \mathsf{PRF}^{\mathsf{mod}}_{(k,a)}, z_0 = 0^{\sqrt{N}/2}, z_1 = e_a\right] \geq 0.1,$$

where e_a denotes the string of length $\sqrt{N}/2$ that is all zero except for the a-th bit.

Analysis of distributions $A_i^{(f,z)}$. By a result of Sahai and Vadhan [51], there are two polynomial-time samplable distributions $A_0^{(f,z)}$ and $A_1^{(f,z)}$ such that $\|A_0^{(f,z)} - A_1^{(f,z)}\| \geq 1 - 2^{-n}$ when $\Pi^{(f,z)}() = 1$; and $\|A_0^{(f,z)} - A_1^{(f,z)}\| \leq 2^{-n}$ when $\Pi^{(f,z)}() = 0$.

Hence, with probability 0.1 over $(k,a) \leftarrow \mathcal{K}^{\mathsf{mod}}$, we have

$$\|A_0^{(f,z_1)} - A_1^{(f,z_1)}\| \geq 1 - 2^{-n} \text{ and } \|A_0^{(f,z_0)} - A_1^{(f,z_0)}\| \leq 2^{-n}.$$

This means that either $\|A_0^{(f,z_0)} - A_0^{(f,z_1)}\| \geq 1/3$ or $\|A_1^{(f,z_0)} - A_1^{(f,z_1)}\| \geq 1/3$.

Now we show that the above implies an algorithm that breaks $\mathsf{PRF}^{\mathsf{mod}}$, and therefore contradicts Lemma 38.

The algorithm and its analysis. Given oracle access to a function $f \leftarrow \mathsf{PRF}^{\mathsf{mod}}_{\mathcal{K}^{\mathsf{mod}}}$, our algorithm first picks a random index $i \in \{0,1\}$. It then simulates A_i with oracle access to (f,z) to take a sample from $A_i^{(f,z)}$, where $z = z_0 = 0^{\sqrt{N}/2}$; it records all the indices in z that are queried by A_i. Now, with probability at least $0.1/2 = 0.05$, we have $\|A_i^{(f,z_0)} - A_i^{(f,z_1)}\| \geq 1/3$. Since (f,z_0) and (f,z_1) only differ at the a-th index of z, we can see that $A_i^{(f,z_0)}$ must have queried the a-th index of z with probability at least $1/3$.

Hence, with probability at least $0.05/3 = \Omega(1)$, one of the values recorded by our algorithm is a, and in that case our algorithm can find a collision in f easily. However, when f is a truly random function, no algorithm can find a collision with a non-negligible probability. Therefore, this algorithm is a distinguisher between $\mathsf{PRF}^{\mathsf{mod}}$ and a truly random function, contradicting the fact that $\mathsf{PRF}^{\mathsf{mod}}$ is secure by Lemma 38.

Construction of the oracle $\mathcal{O}$. Finally, we are ready to construct our oracle $\mathcal{O}$. We will let $\mathcal{O}$ encode pairs $(f_1,z_1),(f_2,z_2),dotsc$, where f_n is a function from $\mathcal{X}_n$ to $\mathcal{X}_n$ and $z_n \in \{0,1\}^{\sqrt{N}/2}$.

For each n, we draw a random index $i \leftarrow \{0,1\}$, and then draw $(f_n,z_n) \leftarrow \mathcal{D}_n^i$. We set L to be the unary language consisting of all 0^n for which (f_n,z_n) is drawn from $\mathcal{D}_n^1$.

From Lemma 38, a quantum algorithm can distinguish $\mathcal{D}_n^0$ from $\mathcal{D}_n^1$, except with negligible probability, by recovering a. Therefore, by a similar argument as in the proof of Theorem 39, we have $L \in \mathsf{BQP}^{\mathcal{O}}$ with probability 1.

On the other hand, for a protocol Π and a sufficiently large n, either (A) happens, which means that $\Pi^{(f_n,z_n)}$ is invalid with probability 0.05 on input 0^n, or (B) happens, which means that Π cannot distinguish $\mathcal{D}_n^0$ and $\mathcal{D}_n^1$ with a constant probability.

In both cases Π cannot decide whether 0^n belongs to L correctly with bounded error. Hence, again by a similar argument as in the proof of Theorem 39, the probability that Π decides L is 0. And since there are only countably many protocols, we have $L \notin \mathsf{SZK}^{\mathcal{O}}$ with probability 1, which means that $\mathsf{BQP}^{\mathcal{O}} \not\subset \mathsf{SZK}^{\mathcal{O}}$ with probability 1.

Finally, it is easy to see that $\mathcal{O} \in \mathsf{P/poly}$, which completes the proof. ◀

B Missing Proofs in Section 3

We first prove Lemma 13.

Proof of Lemma 13. Let $N = 2^n$ for simplicity and L be a list consisting of N reals: $|\langle u|w\rangle|^2 - 2^{-n}$ for each $w \in \{0,1\}^n$. We sort all reals in L in increasing order, and denote them by $a_1, a_2, dotsc, a_N$. We also let $\Delta = \mathsf{dev}(|u\rangle)$ for brevity.

Then from the definitions of $\mathsf{adv}(|u\rangle)$ and $\mathsf{dev}(|u\rangle)$, we have

$$\sum_{i=1}^{N} a_i = 0,$$

$$\sum_{i=1}^{N} |a_i| = \Delta, \text{ and}$$

$$\mathsf{adv}(|u\rangle) = \frac{1}{2} + \sum_{i=N/2+1}^{N} a_i.$$

Now, let t be the first index such that $a_t \geq 0$. Then we have

$$\sum_{i=t}^{N} a_i = \sum_{i=t}^{N} |a_i| = \frac{\Delta}{2} \qquad \text{and} \qquad \sum_{i=1}^{t-1} a_i = -\sum_{i=1}^{t-1} |a_i| = -\frac{\Delta}{2}.$$

We are going to consider the following two cases.

(i) $t \geq N/2 + 1$. Note that a_i's are increasing and for all $i < t$, $a_i < 0$, we have

$$\sum_{i=1}^{N/2} |a_i| \geq \sum_{i=N/2+1}^{t-1} |a_i|,$$

which means

$$\sum_{i=N/2+1}^{t-1} |a_i| \leq \frac{1}{2} \cdot \sum_{i=1}^{t-1} |a_i| \leq \frac{\Delta}{4}.$$

Therefore,

$$\sum_{i=N/2+1}^{N} a_i \geq \sum_{i=t}^{N} a_i + \sum_{i=N/2+1}^{t-1} a_i \geq \frac{1}{2} + \frac{\Delta}{2} - \frac{\Delta}{4} \geq \frac{1}{2} + \frac{\Delta}{4}.$$

(ii) $t \leq N/2$. In this case, note that we have

$$\sum_{i=N/2+1}^{N} a_i \geq \sum_{i=t}^{N/2} a_i.$$

Therefore,

$$\sum_{i=N/2+1}^{N} a_i \geq \frac{1}{2} \cdot \sum_{i=t}^{N} a_i \geq \frac{\Delta}{4}.$$

Since in both cases we have $\sum_{i=N/2+1}^{N} a_i \geq \frac{\Delta}{4}$, it follows that

$$\mathsf{adv}(|u\rangle) = \frac{1}{2} + \sum_{i=N/2+1}^{N} a_i \geq \frac{1}{2} + \frac{\Delta}{4},$$

which completes the proof. ◀

Now we prove Lemma 14.

Proof of Lemma 14. The random pure state $|u\rangle$ can be generated as follows: draw four i.i.d. reals $x_1, x_2, x_3, x_4 \sim \mathcal{N}(0,1)$, and set

$$|u\rangle = \frac{(x_1 + x_2 i)|0\rangle + (x_3 + x_4 i)|1\rangle}{\sqrt{x_1^2 + x_2^2 + x_3^2 + x_4^2}}.$$

Hence, we have

$$\begin{aligned}
&\mathbb{E}\left[\left||\langle u|0\rangle|^2-|\langle u|1\rangle|^2\right|\right]\\
=&\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\frac{1}{(2\pi)^2}\frac{|x_1^2+x_2^2-x_3^2-x_4^2|}{x_1^2+x_2^2+x_3^2+x_4^2}\cdot e^{-(x_1^2+x_2^2+x_3^2+x_4^2)/2}dx_1dx_2dx_3dx_4\\
=&\int_0^{2\pi}\int_0^{2\pi}\int_0^{+\infty}\int_0^{+\infty}\frac{1}{(2\pi)^2}\cdot\frac{|\rho_1^2-\rho_2^2|}{\rho_1^2+\rho_2^2}\cdot\rho_1\rho_2\cdot e^{-(\rho_1^2+\rho_2^2)/2}d\rho_1d\rho_2d\theta_1d\theta_2\\
&\qquad\qquad(x_1=\rho_1\sin\theta_1,\ y_1=\rho_1\cos\theta_1,\ x_2=\rho_2\sin\theta_2,\ y_2=\rho_2\cos\theta_2)\\
=&\int_0^{+\infty}\int_0^{+\infty}\frac{|\rho_1^2-\rho_2^2|}{\rho_1^2+\rho_2^2}\cdot\rho_1\rho_2\cdot e^{-(\rho_1^2+\rho_2^2)/2}d\rho_1d\rho_2\\
=&\frac{1}{2}
\end{aligned}$$

◀

C Missing Proofs in Section 6

We prove Lemma 30 here.

Proof of Lemma 30. We prove the concentration inequality by bounding the variance,

$$\mathrm{Var}[\mathsf{adv}(f)]=\mathbb{E}[\mathsf{adv}(f)^2]-\mathbb{E}[\mathsf{adv}(f)]^2.$$

Note that

$$\mathbb{E}[\mathsf{adv}(f)]^2=\left(\frac{2}{\sqrt{2\pi}}\int_1^{+\infty}x^2e^{-x^2/2}dx\right)^2=\mathsf{Succ}_Q^2.$$

We now calculate $\mathbb{E}[\mathsf{adv}(f)^2]$. We have

$$\begin{aligned}
\mathop{\mathbb{E}}_f[\mathsf{adv}(f)^2]&=\mathop{\mathbb{E}}_f\left[\left(\mathop{\mathbb{E}}_{z\in\{0,1\}^n}[\widehat{f}^2(z)\cdot\mathbf{1}_{|\widehat{f}(z)|\geq 1}]\right)^2\right]\\
&=\mathop{\mathbb{E}}_f\left[\mathop{\mathbb{E}}_{z_1,z_2\in\{0,1\}^n}\left[\widehat{f}^2(z_1)\widehat{f}^2(z_2)\cdot\mathbf{1}_{|\widehat{f}(z_1)|\geq 1\wedge|\widehat{f}(z_2)|\geq 1}\right]\right].\\
&=\mathop{\mathbb{E}}_{z_1,z_2\in\{0,1\}^n}\left[\mathop{\mathbb{E}}_f\left[\widehat{f}^2(z_1)\widehat{f}^2(z_2)\cdot\mathbf{1}_{|\widehat{f}(z_1)|\geq 1\wedge|\widehat{f}(z_2)|\geq 1}\right]\right].
\end{aligned}$$

Now there are two cases: $z_1=z_2$ and $z_1\neq z_2$. When $z_1=z_2$, let $z=z_1=z_2$; then we have

$$\begin{aligned}
Ex_f\left[\widehat{f}^2(z_1)\widehat{f}^2(z_2)\cdot\mathbf{1}_{|\widehat{f}(z_1)|\geq 1\wedge|\widehat{f}(z_2)|\geq 1}\right]&=\mathop{\mathbb{E}}_f\left[\widehat{f}^4(z)\cdot\mathbf{1}_{|\widehat{f}(z)|\geq 1}\right]\\
&=\frac{2}{\sqrt{2\pi}}\int_1^{+\infty}x^4e^{-x^2/2}dx\\
&=O(1).
\end{aligned}$$

Next, if $z_1\neq z_2$, then without loss of generality, we can assume $z_1=0^N$. Now we define two sets A and B,

$$A=\{x\in\{0,1\}^n:(z_2\cdot x)=0\}\text{ and }B=\{x\in\{0,1\}^n:(z_2\cdot x)=1\}.$$

We also define

$$\widehat{f}_A:=\frac{1}{\sqrt{N/2}}\cdot\sum_{z\in A}f(z)\text{ and }\widehat{f}_B:=\frac{1}{\sqrt{N/2}}\cdot\sum_{z\in B}f(z).$$

Then from the definitions of $\widehat{f}(z_1)$ and $\widehat{f}(z_2)$, we have

$$\widehat{f}(z_1) = \frac{1}{\sqrt{2}} \cdot (\widehat{f}_A + \widehat{f}_B) \text{ and } \widehat{f}(z_1) = \frac{1}{\sqrt{2}} \cdot (\widehat{f}_A - \widehat{f}_B).$$

Therefore,

$$\mathop{\mathbb{E}}_{f}\left[\widehat{f}^2(z_1)\widehat{f}^2(z_2) \cdot \mathbf{1}_{|\widehat{f}(z_1)|\geq 1 \wedge |\widehat{f}(z_2)|\geq 1}\right] = \left(\frac{1}{\sqrt{2\pi}}\right)^2 \cdot \int_{\substack{|a+b|\geq\sqrt{2} \\ |a-b|\geq\sqrt{2}}} \frac{1}{4} \cdot (a+b)^2 \cdot (a-b)^2 \cdot e^{-(a^2+b^2)/2} \cdot dadb.$$

Let $x = a + b$ and $y = a - b$. Then

$$a = \frac{x+y}{2}, \quad b = \frac{x-y}{2}, \quad da = \frac{dx+dy}{2}, \quad \text{and} \quad db = \frac{dx-dy}{2}.$$

Also note that $x^2 + y^2 = 2(a^2 + b^2)$. Plugging in x and y, the above can be simplified to

$$\begin{aligned}
\frac{1}{2\pi}\int_{\substack{|x|\geq\sqrt{2} \\ |y|\geq\sqrt{2}}} \frac{1}{4}x^2y^2e^{-(x^2+y^2)/4} \cdot \frac{1}{2}dxdy &= \frac{1}{2\pi}\left(\int_{|x|\geq\sqrt{2}} \frac{1}{2\sqrt{2}} \cdot x^2e^{-x^2/4}dx\right)^2 \\
&= \frac{1}{2\pi}\left(\int_{\sqrt{2}}^{+\infty} \frac{1}{\sqrt{2}} \cdot x^2e^{-x^2/4}dx\right)^2 \\
&= \frac{1}{2\pi}\left(\int_{1}^{+\infty} 2t^2e^{-t^2/2}dt\right)^2 && (t = x/\sqrt{2}) \\
&= \left(\frac{2}{\sqrt{2\pi}}\int_{1}^{+\infty} t^2e^{-t^2/2}dt\right)^2 = && \mathsf{Succ}_Q^2.
\end{aligned}$$

Putting two cases together, we have

$$\mathop{\mathbb{E}}_{f}[\mathsf{adv}(f)^2] = \frac{1}{N} \cdot O(1) + \frac{N-1}{N} \cdot \mathsf{Succ}_Q^2,$$

which in turn implies

$$\mathrm{Var}[\mathsf{adv}(f)] = O(1/N).$$ ◀

D Missing Proofs in Section 7

For completeness, we prove Lemma 38 here.

Proof of Lemma 38. In the following, we will always use $\varepsilon = \varepsilon(n)$ to denote a negligible function. And we will denote $\mathcal{X}^{\mathsf{raw}}$ as $\mathcal{X}$ for brevity. Recall that we interpret $\mathcal{X}$ as $[N]$ for $N = N(n) = \mathcal{X}$.

Both $\mathsf{PRP}^{\mathsf{raw}}$ and $\mathsf{PRF}^{\mathsf{mod}}$ are classically-secure PRFs. It is well-known that a secure PRP is also a secure PRF; therefore $\mathsf{PRP}^{\mathsf{raw}}$ is a classically-secure PRF. So we only need to prove this for $\mathsf{PRF}^{\mathsf{mod}}$.

Recall that $\mathsf{PRF}^{\mathsf{mod}}_{(k,a)}(x) = \mathsf{PRP}^{\mathsf{raw}}_k((x-1) \bmod a + 1)$. We first show that if the $\mathsf{PRP}^{\mathsf{raw}}$ in the definition of $\mathsf{PRF}^{\mathsf{mod}}$ were replaced by a truly random function, then no classical polynomial-time algorithm A could distinguish it from a truly random function. That is,

$$\left|\Pr_{f\leftarrow\mathcal{X}^{\mathcal{X}},a\leftarrow\mathcal{A}}[A^{f\bmod a}() = 1] - \Pr_{f\leftarrow\mathcal{X}^{\mathcal{X}}}[A^{f}() = 1]\right| < \varepsilon, \tag{23}$$

where $f_{\bmod a}(x) := f((x-1) \bmod a + 1)$.

Clearly, as long as A never queries its oracle on two points x and x' such that $x \equiv x' \pmod{a}$, the oracle will look random. Suppose A makes q queries in total. There are $\binom{q}{2}$ possible differences between query points, and each difference is at most N. So for large enough N, each difference can be divisible by at most two different moduli from $\mathcal{A}$ (recall that each number in $\mathcal{A}$ lies in $[\sqrt{N}/4, sqrtN/2]$). And since $|\mathcal{A}| \geq \Omega(\sqrt{N}/\log N)$, the total probability of querying two x and x' such that $x \equiv x' \pmod{a}$ is at most

$$O\left(\frac{q^2 \log N}{\sqrt{N}}\right),$$

which is negligible as N is exponential in n. This implies (23).

Now, since $\mathsf{PRP}^{\mathsf{raw}}$ is a classically-secure PRF, for any polynomial-time algorithm A, we have

$$\left| \Pr_{f \leftarrow \mathcal{X}^{\mathcal{X}}, a \leftarrow \mathcal{A}} [A^{f \bmod a}() = 1] - \Pr_{f \leftarrow \mathsf{PRP}^{\mathsf{raw}}_{\mathcal{K}^{\mathsf{raw}}}, a \leftarrow \mathcal{A}} [A^{f \bmod a}() = 1] \right| < \varepsilon, \tag{24}$$

since otherwise we can directly construct a distinguisher between $\mathsf{PRP}^{\mathsf{raw}}_{\mathcal{K}^{\mathsf{raw}}}$ and $\mathcal{X}^{\mathcal{X}}$.

Note that

$$\Pr_{f \leftarrow \mathsf{PRP}^{\mathsf{raw}}_{\mathcal{K}^{\mathsf{raw}}}, a \leftarrow \mathcal{A}} [A^{f \bmod a}() = 1] = \Pr_{f \leftarrow \mathsf{PRF}^{\mathsf{mod}}_{\mathcal{K}^{\mathsf{mod}}}} [A^{f}() = 1]$$

by their definitions. Hence, (23) and (24) together imply that

$$\left| \Pr_{f \leftarrow \mathsf{PRF}^{\mathsf{mod}}_{\mathcal{K}^{\mathsf{mod}}}} [A^{f}() = 1] - \Pr_{f \leftarrow \mathcal{X}^{\mathcal{X}}} [A^{f}() = 1] \right| < \varepsilon$$

for any polynomial-time algorithm A. This completes the proof for the first statement.

Quantum algorithm for recovering a given oracle access to $\mathsf{PRF}^{\mathsf{mod}}_{\mathcal{K}^{\mathsf{mod}}}$. Let $(k, a) \leftarrow \mathcal{K}^{\mathsf{mod}}$, $f = \mathsf{PRF}^{\mathsf{mod}}_{(k,a)}$ and $g = \mathsf{PRP}^{\mathsf{raw}}_{k}$. From the definitions, we have $f = g \bmod a$.

Since g is a permutation, there is no collision (x, x') such that $g(x) = g(x')$. Moreover, in this case, $f = g \bmod a$ has a unique period a. Therefore, we can apply Boneh and Lipton's quantum period-finding algorithm [20] to recover a. Using a polynomial number of repetitions, we can make the failure probability negligible, which completes the proof for the second statement.

Quantum algorithm for distinguishing $\mathsf{PRP}^{\mathsf{raw}}$ and $\mathsf{PRF}^{\mathsf{mod}}$. Finally, we show the above algorithm implies a good quantum distinguisher between $\mathsf{PRP}^{\mathsf{raw}}$ and $\mathsf{PRF}^{\mathsf{mod}}$. Given oracle access to a function f, our distinguisher A tries to recover a period a using the previously discussed algorithm, and accepts only if $f(1) = f(1 + a)$.

When $f \leftarrow \mathsf{PRP}^{\mathsf{raw}}_{\mathcal{K}^{\mathsf{raw}}}$, note that f is a permutation, which means A accepts with probability 0 in this case.

On the other side, when $f \leftarrow \mathsf{PRF}^{\mathsf{mod}}_{\mathcal{K}^{\mathsf{mod}}}$, from the second statement, A can recover the period a with probability at least $1 - \varepsilon$. Therefore A accepts with probability at least $1 - \varepsilon$.

Combining, we find that A is a distinguisher with advantage $1 - \varepsilon$, and this completes the proof for the last statement. ◀

E Numerical Simulation For Conjecture 1

Recall Conjecture 1, which said that a random quantum circuit C on n qubits satisfies $\mathsf{adv}(C) \geq C_{\mathsf{thr}} - \varepsilon$ with probability $1 - 1/\exp(n)$, where

$$C_{\mathsf{thr}} := \frac{1+\ln 2}{2}.$$

We first explain where the magic number C_{thr} comes from. Suppose C is drawn from $\mu_{\mathsf{Haar}}^{2^n}$ instead of μ_{grid}. Then $C|0^n\rangle$ is a random quantum state, and therefore the values $2^n \cdot |\langle x|C|0\rangle|$'s, for each $x \in \{0,1\}^n$ are distributed very closely to 2^n i.i.d. exponential distributions with $\lambda = 1$.

So, assuming that, we can see that the median of $\mathsf{probList}(C|0\rangle)$ concentrates around $\ln 2$, as

$$\int_0^{\ln 2} e^{-x} dx = \frac{1}{2},$$

which also implies that $\mathsf{adv}(C)$ concentrates around

$$\int_{\ln 2}^{+\infty} x e^{-x} dx = C_{\mathsf{thr}} = \frac{1+\ln 2}{2} \approx 0.846574.$$

In the following, we first provide some numerical evidence that the values in $\mathsf{probList}(C|0\rangle)$ *also* behave like exponentially distributed random variables, which explains why the constant should indeed be C_{thr}. Then we provide a direct numerical simulation for the distribution of $\mathsf{adv}(C)$ to argue that $\mathsf{adv}(C)$ approximately follows a nice normal distribution. Finally we examine the decreasing rate of the standard variance of $\mathsf{adv}(C)$ to support our conjecture.

E.1 Numerical Simulation Setting

In the following we usually set $n = 9$ or $n = 16$ (so that $\sqrt{n}$ is an integer); and we always set $m = n^2$ as in Conjecture 1.

E.2 Distribution of probList($C|0\rangle$): Approximate Exponential Distribution

In Figure 2 we plot the histogram of the distribution of the normalized probabilities in $\mathsf{probList}(C|0\rangle)$ where $C \leftarrow \mu_{\mathsf{grid}}^{16,256}$, that is,

$$\{2^n \cdot p : p \in \mathsf{probList}(C|0\rangle)\}.$$

And we compare it with the exponential distribution with $\lambda = 1$. From Figure 2, it is easy to observe that these two distributions are quite similar.

E.3 Distribution of adv(C): Approximate Normal Distribution

Next we perform direct numerical simulation to see how $\mathsf{adv}(C)$ is distributed when $C \leftarrow \mu_{\mathsf{grid}}^{n,m}$. Our results suggest that $\mathsf{adv}(C)$ approximately follows a normal distribution with mean close to C_{thr}.

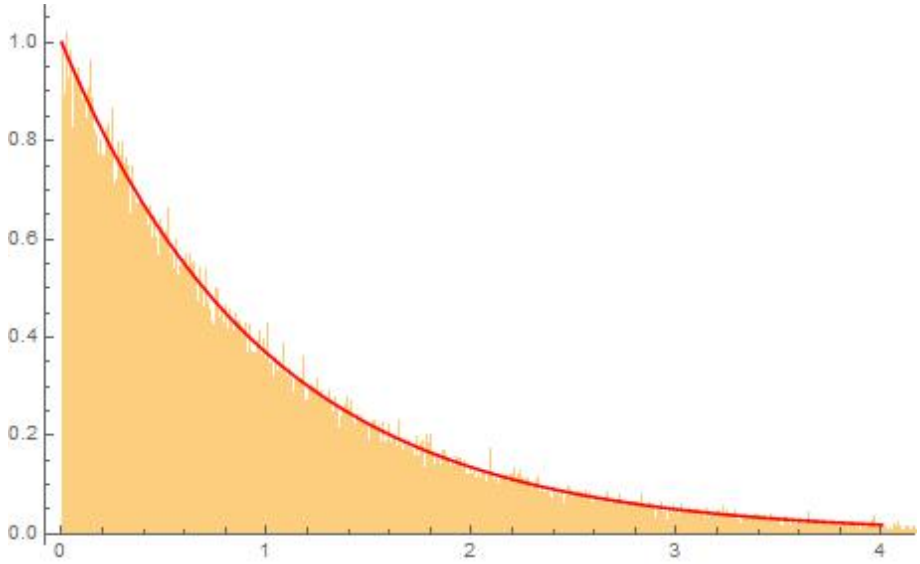


Figure 2 A histogram of (normalized) $\mathsf{probList}(C|0\rangle)$, where $C \leftarrow \mu_{\mathsf{grid}}^{16,256}$. The x-axis represents the probability, and the y-axis represents the estimated density, and the red line indicates the PDF of the exponential distribution with $\lambda = 1$.

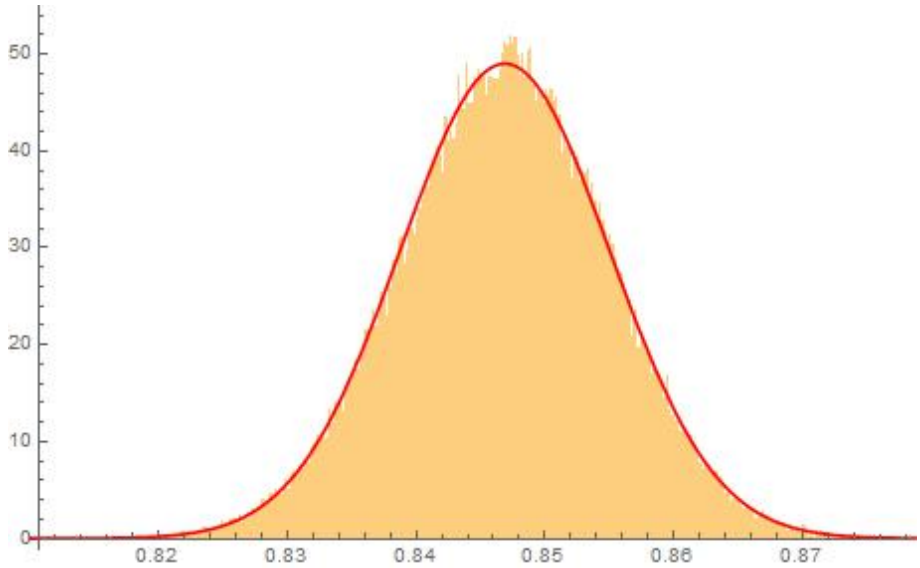


Figure 3 A histogram of the $\mathsf{adv}(C)$'s of the 10^5 i.i.d. samples from $\mu_{\mathsf{grid}}^{9,81}$. The x-axis represents the value of $\mathsf{adv}(C)$, and the y-axis represents the estimated density, and the red line indicates the PDF of the normal distribution $\mathcal{N}(0.846884, 0.00813911^2)$.

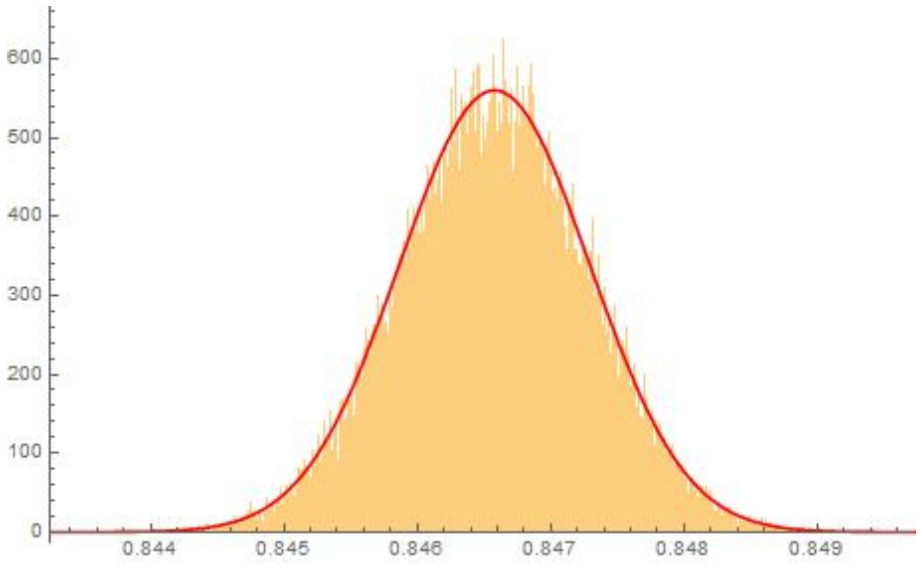


Figure 4 A histogram of the $\mathsf{adv}(C)$'s of the 10^5 i.i.d. samples from $\mu_{\mathsf{grid}}^{16,256}$. The x-axis represents the value of $\mathsf{adv}(C)$, the y-axis represents the estimated density, and the red line indicates the PDF of the normal distribution $\mathcal{N}(0.846579, 0.000712571^2)$.

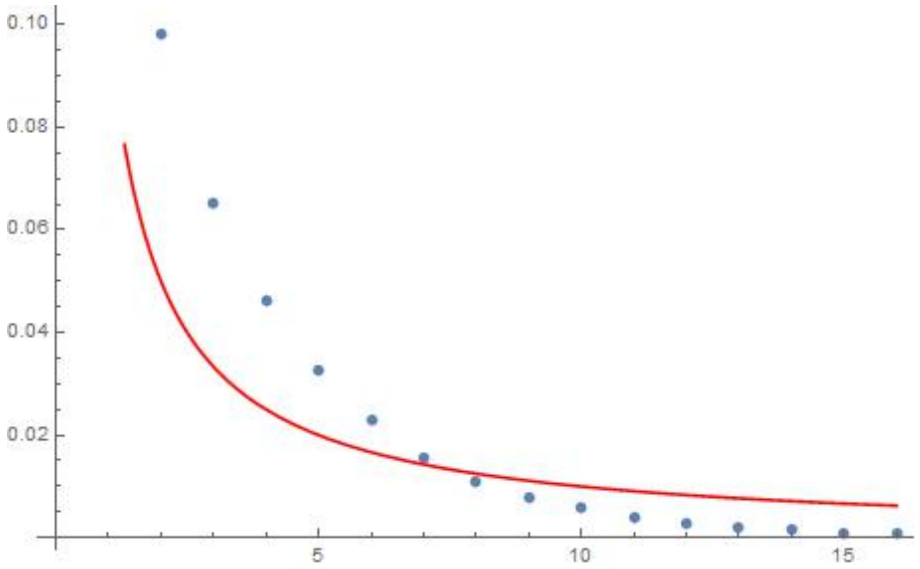


Figure 5 The empirical decay of the variance of $\mathsf{adv}(C)$. Here a point (x, y) means that the standard variance of the corresponding $\mathsf{adv}(C)$'s for the 1000 i.i.d. samples from $\mu_{\mathsf{general}}^{x,x^2}$ is y. Also, the red line represents the function $y = 0.1/x$.

E.3.1 $\mu_{\mathsf{grid}}^{9,81}$, 10^5 samples

We first draw 10^5 i.i.d. samples from $\mu_{\mathsf{grid}}^{9,81}$ and plot the distribution of the corresponding $\mathsf{adv}(C)$'s in Figure 3. From Figure 3, we can see that the distribution of $\mathsf{adv}(C)$ follows a nice normal distribution, with mean very close to C_{thr}.

E.3.2 $\mu_{\mathsf{grid}}^{16,256}$, 10^5 samples

Next, we draw 10^5 i.i.d. samples from $\mu_{\mathsf{grid}}^{16,256}$ and plot the distribution of the corresponding $\mathsf{adv}(C)$'s in Figure 4. From Figure 4, we can observe that the distribution of $\mathsf{adv}(C)$ in this case also mimics a nice normal distribution, with mean even closer to C_{thr} than in the previous case.

E.4 The Empirical Decay of Variance

The previous subsection suggests that $\mathsf{adv}(C)$ follows a normal distribution with mean approaching C_{thr}. If that's indeed the case, then informally, Conjecture 1 becomes equivalent to the conjecture that the variance σ of C_{thr} becomes $O(1/n)$ as $n \to +\infty$. So we wish to verify the latter conjecture for $\mu_{\mathsf{grid}}^{n,n^2}$ with some numerical simulation.

The circuit distribution $\mu_{\mathsf{general}}^{n,m}$

Unfortunately, the definition of $\mu_{\mathsf{grid}}^{n,m}$ requires n to be a perfect square, and there are only five perfect squares for which we can perform quick simulations ($n \in \{1, 4, 9, 16, 25\}$). So we consider the following distribution $\mu_{\mathsf{general}}^{n,m}$ on n qubits and m circuits instead: each of m gates is a Haar random two-qubit gate acting on two qubits chosen uniformly at random. In this case, since we don't need to arrange the qubits in a square grid, n can be any positive integer.

Numerical simulation shows that $\mathsf{adv}(C)$ is distributed nearly the same when C is drawn from $\mu_{\mathsf{general}}^{n,n^2}$ or $\mu_{\mathsf{grid}}^{n,n^2}$ for $n = 3$ or $n = 4$, so it is reasonable to consider μ_{general} instead of μ_{grid}.

For each $n = 2, 3, dotsc, 16$, we draw 1000 i.i.d. samples from $\mu_{\mathsf{general}}^{n,n^2}$, and calculate the variance of the corresponding $\mathsf{adv}(C)$'s. The results are summarized in Figure 5.

From Figure 5, we can observe that the variance decreases faster than the inverse function $1/x$; hence it supports Conjecture 1.

Augmented Index and Quantum Streaming Algorithms for DYCK(2)*†

Ashwin Nayak[1] and Dave Touchette[2]

1 **Department of Combinatorics and Optimization, and Institute for Quantum Computing, University of Waterloo, Waterloo, ON, Canada**
ashwin.nayak@uwaterloo.ca

2 **Department of Combinatorics and Optimization, and Institute for Quantum Computing, University of Waterloo, Waterloo, ON, Canada; and Perimeter Institute for Theoretical Physics, Waterloo, ON, Canada**
touchette.dave@gmail.com

Abstract

We show how two recently developed quantum information theoretic tools can be applied to obtain lower bounds on quantum information complexity. We also develop new tools with potential for broader applicability, and use them to establish a lower bound on the quantum information complexity for the Augmented Index function on an easy distribution. This approach allows us to handle superpositions rather than distributions over inputs, the main technical challenge faced previously. By providing a quantum generalization of the argument of Jain and Nayak [IEEE TIT'14], we leverage this to obtain a lower bound on the space complexity of multi-pass, unidirectional quantum streaming algorithms for the DYCK(2) language.

1998 ACM Subject Classification E.4 Coding and Information Theory, F.1.3 Complexity Measures and Classes, F.2.3 Tradeoffs between Complexity Measures

Keywords and phrases Quantum streaming algorithms, Space complexity, Quantum communication complexity, Quantum information cost, DYCK(2), Augmented Index

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.23

1 Introduction

The first *bona fide* quantum computers that are built are likely to involve a few hundred qubits, and be limited to short computations. This prompted much research into the capabilities of space bounded quantum computation, especially of quantum finite automata, during the early development of the theory of quantum computation (see, e.g., Refs. [21, 16, 1, 2]). More recently, this has led to the investigation of *quantum streaming algorithms* [18, 11, 5, 20].

1.1 Streaming Algorithms and Augmented Index

Streaming algorithms were originally proposed as a means to process massive real-world data that cannot be stored in their entirety in computer memory [22]. Instead of random access to the input data, these algorithms receive the input in the form of a *stream*, i.e., one input symbol at a time. The algorithms attempt to solve some information processing task using

* A full version of the paper is available at https://arxiv.org/abs/1610.04937.

† A.N.'s research is supported in part by NSERC Canada. D.T.'s research is supported in part by NSERC, CIFAR and Industry Canada. IQC and PI are supported in part by the Government of Canada and the Province of Ontario.

as little space and time as possible, on occasion using more than one sequential pass over the input stream.

Streaming algorithms provide a natural framework for studying simple, small-space quantum computation beyond the scope of quantum finite automata. Some of the works mentioned above (e.g., LeGall [18]) show how quantum streaming algorithms can accomplish certain specially crafted tasks with exponentially smaller space, as compared with classical algorithms. This led Jain and Nayak [12] to ask how much more efficient such quantum algorithms could be for other, more natural problems. They focused on DYCK(2), a well-studied and important problem from formal language theory [8]. DYCK(2) consists of all well-formed expressions with two types of parenthesis, denoted below by $a, \overline{a}$ and $b, \overline{b}$, with the bar indicating a closing parenthesis. More formally, DYCK(2) is the language over the alphabet $\Sigma = \{a, \overline{a}, b, \overline{b}\}$ defined recursively as

$$\text{DYCK}(2) \quad = \quad \varepsilon + \left(a \cdot \text{DYCK}(2) \cdot \overline{a} + b \cdot \text{DYCK}(2) \cdot \overline{b}\right) \cdot \text{DYCK}(2) \ ,$$

where ε is the empty string, '$\cdot$' indicates concatenation of strings (or subsets thereof) and '$+$' denotes set union.

The related problem of recognizing whether a given expression with parentheses is well-formed was originally studied by Magniez, Mathieu, and Nayak [19] in the context of *classical* streaming algorithms. They discovered a remarkable phenomenon, that providing *bi-directional* access to the input stream leads to an exponentially more space-efficient algorithm. They presented a randomized streaming algorithm that makes one pass over the input, uses $O(\sqrt{n \log n})$ bits, and makes polynomially small probability of error to determine membership of expressions of length $O(n)$ in DYCK(2). Moreover, they proved that this space bound is optimal for error at most $1/(n \log n)$, and conjectured that a similar polynomial space bound holds for multi-pass algorithms as well. Magniez *et al.* complemented this with a second randomized algorithm that makes two passes in *opposite* directions over the input, uses only $O(\log^2 n)$ space, and has polynomially small probability of error. Later, two sets of authors [6, 12] independently and concurrently proved the conjectured hardness of DYCK(2) for classical multi-pass streaming algorithms that read the input only in one direction. They showed that any unidirectional randomized T-pass streaming algorithm that recognizes length n instances of DYCK(2) with a constant probability of error uses space $\Omega(\sqrt{n}/T)$.

The space lower bounds for DYCK(2) established in Refs. [19, 6, 12] all rely on a connection with a two-party communication problem, Augmented Index, a variant of Index, a basic problem in two-party communication complexity. In the Index function problem, one party, Alice, is given a string $x \in \{0,1\}^n$, and the other party, Bob, is given an index $k \in [n]$, for some positive integer n. Their goal is to communicate with each other and compute x_k, the kth bit of the string x. In the Augmented Index function problem, Bob is given the prefix $x[1, k-1]$ (the first $k-1$ bits of x) and a bit b in addition to the index k. The goal of the two parties is to determine if $x_k = b$ or not. The three works cited above (see also [7]) all prove information cost trade-offs for Augmented Index. As a result, in any bounded-error protocol for the function, either Alice reveals $\Omega(n)$ information about her input x, or Bob reveals $\Omega(1)$ information about the index k (even under an easy distribution, the uniform distribution over zeros of the function).

Motivated by the abovementioned works, Jain and Nayak [12] studied quantum protocols for Augmented Index. They defined a notion of quantum information cost for distributions with a limited form of dependence, and then arrived at a similar tradeoff as in the classical case. This result, however, does not imply a lower bound on the space required by quantum

streaming algorithms for DYCK(2). The issue is that the reduction from low information cost protocols for Augmented Index to small space streaming algorithms breaks down in the quantum case (for the notion of quantum information cost they proposed). This left open the possibility of more efficient unidirectional quantum streaming algorithms.

1.2 Overview of Results

We establish the following lower bound on the space complexity of T-pass, unidirectional quantum streaming algorithms for DYCK(2), thus solving the question posed by Jain and Nayak [12].

▶ **Theorem 1.** *For any $T \geq 1$, any unidirectional T-pass quantum streaming algorithm that recognizes* DYCK(2) *with a constant probability of error uses space $\Omega(\sqrt{n}/T^3)$ on length n instances of the problem.*

The space bound above holds for a general model for quantum streaming algorithms, one in which the computation is characterized by arbitrary quantum operations. In particular, the computation may be non-unitary, and may use "on-demand" ancillary qubits in addition to the allowed work space. Some earlier work showing strong limitations of bounded space, such as that on quantum finite automata [2], assumed unitary evolution.

Theorem 1 shows that, possibly up to logarithmic factors and the dependence on the number of passes, quantum streaming algorithms are no more efficient than classical ones for this problem. In particular, this provides the first natural example for which classical bi-directional streaming algorithms perform exponentially better than unidirectional quantum streaming algorithms.

Theorem 1 is a consequence of a lower bound that we establish on a measure of quantum information cost introduced by Touchette [25]. (Henceforth, we use the term "quantum information cost" without any qualification to refer to this notion.) We consider this cost for any quantum protocol Π computing the Augmented Index function, with respect to an "easy" distribution μ_0: the uniform distribution over the zeros of the function. Due to the asymmetry of the Augmented Index function, we distinguish between the amount of information Alice transmits to Bob, denoted $\mathrm{QIC}_{\mathrm{A}\to\mathrm{B}}(\Pi, \mu_0)$ and the amount of information Bob transmits to Alice, denoted $\mathrm{QIC}_{\mathrm{B}\to\mathrm{A}}(\Pi, \mu_0)$; see Section 2.3 for formal definitions for these notions. Our main technical contributions are in proving the following trade-off.

▶ **Theorem 2.** *In any t-round quantum protocol Π computing the Augmented Index function f_n with constant error $\varepsilon \in [0, 1/4)$ on any input, either $\mathrm{QIC}_{\mathrm{A}\to\mathrm{B}}(\Pi, \mu_0) \in \Omega(n/t^2)$ or $\mathrm{QIC}_{\mathrm{B}\to\mathrm{A}}(\Pi, \mu_0) \in \Omega(1/t^2)$.*

A more precise statement is presented as Theorem 17. As in previous works, establishing a lower bound on the quantum information cost for such an easy distribution is necessary; the direct sum argument that links quantum streaming algorithms to quantum protocols for Augmented Index crucially hinges on this. (This phenomenon is common in such direct sum arguments.)

The high level intuition underlying the proof of Theorem 2 and its structure is the same as that in Ref. [12]. There are two principal challenges in their approach, and the choice of an appropriate measure of information cost is fundamental to overcoming both challenges. The first challenge is a direct sum argument that relates streaming algorithms for DYCK(2) and communication protocols for Augmented Index. The second challenge is establishing an information cost trade-off for Augmented Index. Jain and Nayak considered several notions of information cost, each one of which was effective in addressing one challenge *but*

not the other. This was further complicated by the intrinsic correlation of the inputs for Augmented Index held by the two parties. Indeed, an important motivation behind the notion of quantum information cost used in Ref. [12] is the desire to avoid leaking information about the inputs by virtue of their preparation in superposition, instead of exchanging information through interaction alone. The notion they analyzed in detail admits an information cost trade-off, but not a connection between streaming algorithms and low information protocols. In particular, the notion does not seem to be bounded by communication complexity.

Quantum information cost, as proposed by Touchette [25], turns out to be a suitable choice for quantifying the information content of messages in our context. It is defined in terms of conditional mutual information, conditioned on the recipient's quantum state. Thus, this notion naturally avoids the difficulties arising from the intrinsic correlation between the two parties' inputs. It is also relatively simple to derive low quantum information cost protocols for Augmented Index from small-space streaming algorithms for DYCK(2), through a direct sum argument. Remarkably, the properties of quantum information cost allow us to execute the reduction even for algorithms whose computation involves arbitrary quantum operations, including non-unitary evolution. However, a quantum information cost trade-off for Augmented Index still presents significant obstacles. In order to overcome these, we develop new tools for quantum communication complexity that we believe have broader applicability.

One tool is a generalization of the well-known Average Encoding Theorem of (classical and) quantum complexity theory [15], which formalizes the intuition that weakly correlated systems are nearly independent. We call this generalized version the *Superposition-Average Encoding Theorem*, as it allows us to handle arbitrary superpositions over inputs to quantum communication protocols (as opposed to classical distributions over inputs). The proof of this theorem builds on the breakthrough result by Fawzi and Renner [9], linking conditional quantum mutual information to the optimal recovery map acting on the conditioning system. Note that there is an obvious generalization of the Average Encoding Theorem to an analogous result for conditional quantum mutual information implied by the Fawzi-Renner inequality together with the Uhlmann theorem. This *cannot* directly be used in a proof *à la* Ref. [12]. For one, such a generalization would give us a unitary operation that acts on one part of a (pure) "reconstructed" state, and maps it to a state close to a target state. The hybrid argument in Ref. [12] relies on the commutativity of such unitary operations corresponding to successive messages in a protocol, whereas the operations do not commute.

Another key ingredient in the proof of Theorem 2 is a *Quantum Cut-and-Paste Lemma*, a variant of a technique used in Refs. [13, 12], that allows us to deal with easy distributions over inputs. The cut-and-paste lemma for randomized communication protocols connects the distance between transcripts obtained by running protocols on inputs chosen from a two-by-two rectangle $\{x, x'\} \times \{y, y'\}$. The cut-and-paste lemma is very powerful, and a direct quantum analogue does not hold. We can nevertheless obtain the following weaker variant, linking any four possible pairs of inputs in a two-by-two rectangle: if the states for a fixed input y to Bob are close up to a local unitary operator on Alice's side and the states for a fixed input x to Alice are close up to a local unitary operator on Bob's side, then, up to local unitary operators on Alice's and Bob's sides, the states for all pairs (x'', y'') of inputs in the rectangle $\{x, x'\} \times \{y, y'\}$ are close to each other. This lemma allows us to link output states of protocols on inputs from an easy distribution, all mapping to the same output value, to an output state corresponding to a different output value. This helps derive a contradiction to the assumption of low quantum information cost, as states corresponding to different outputs are distinguishable with constant probability.

We go a step further with the quantum information cost trade-off. We provide an alternative way to achieve a similar result, by using a notion of information cost tailored to the Augmented Index problem. An important stepping stone in this approach is the recently developed *Information Flow Lemma* due to Laurière and Touchette [17]. The lemma allows us to track the transfer of information due to interaction in quantum protocols, and provides insight into how information might be leaked due to a superposition over inputs. By conditioning on a suitable classical register, we avoid such leakage of information. Pushing these ideas further, we are able to bring the Average Encoding Theorem to bear in this context as well. This helps us obtain a slightly better round-dependence in the information cost trade-off.

Organization

Background and definitions related to quantum communication, information theory, and streaming algorithms are presented in Section 2. We then adapt, in Section 3, the known two-step reduction from Augmented Index to Dyck(2) to the new notion for information cost due to Touchette [25] and to the general model for streaming algorithms that we define. The main technical tools that we develop and use are presented in Section 4. The main lower bound on the quantum information cost of the Augmented Index function is derived in Section 5. A lower bound with a slightly better dependence on the number of rounds is presented in Section 6.

2 Preliminaries

The full version of this work [23] contains a more detailed preliminaries section, in particular with additional details about the communication model and the properties of the distance and information measures that are relevant for our purposes.

2.1 Quantum Communication Complexity

We refer the reader to text books such as [26, 27] for standard concepts and the associated notation from quantum information.

The notation we use for interactive communication between two parties, called Alice and Bob by convention, is summarized in Figure 1. The operations $U_1, \ldots, U_{M+1}$ in protocol Π are isometries.

We restrict our attention to protocols with classical inputs XY, with $A_{\text{in}}B_{\text{in}}$ initialized to XY, and to so-called "safe protocols". Safe protocols only use $A_{\text{in}}B_{\text{in}}$ as control registers. As explained in Section 2.3, this does not affect the results presented in this article.

We imagine that the joint classical input XY is purified by a register R. We often partition the purifying register as $R = R_X R_Y$, indicating that the classical input XY, distributed as ν, and represented by the quantum state ρ_ν:

$$\rho_\nu^{XY} \quad = \quad \sum_{x,y} \nu(x,y) \ |x\rangle\langle x|^X \otimes |y\rangle\langle y|^Y \tag{1}$$

is purified as

$$|\rho_\nu\rangle \quad = \quad \sum_{x,y} \sqrt{\nu(x,y)} \ |xxyy\rangle^{XR_XYR_Y} \quad . \tag{2}$$

We also use other partitions more appropriate for our purposes, corresponding to particular preparations of the inputs X and Y.

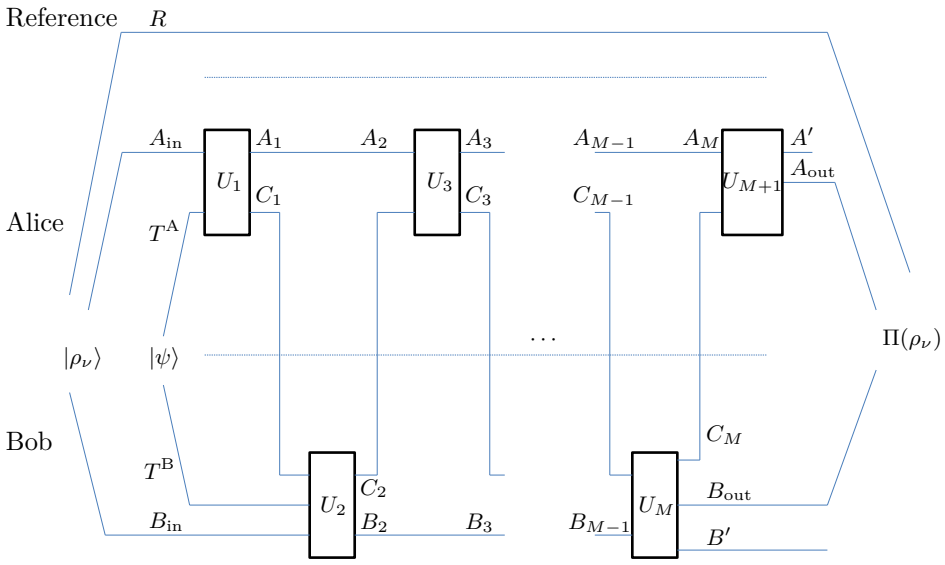


Figure 1 Depiction of an interactive quantum protocol, adapted from Ref. [24, Figure 1], the full version of Ref. [25].

We define the quantum communication cost of Π from Alice to Bob as

$$\mathrm{QCC}_{\mathrm{A}\to\mathrm{B}}(\Pi) \quad := \quad \sum_{0\le i\le (M-1)/2} \log |C_{2i+1}| \ , \tag{3}$$

and the quantum communication cost of Π from Bob to Alice as

$$\mathrm{QCC}_{\mathrm{B}\to\mathrm{A}}(\Pi) \quad := \quad \sum_{1\le i\le M/2} \log |C_{2i}| \ , \tag{4}$$

where for a register D, the notation $|D|$ stands for the dimension of the state space associated with the register. The total communication cost of the protocol is then the sum of these two quantities.

2.2 Information Theory

In order to distinguish between quantum states, we use two related distance measures: trace distance and Bures distance.

The trace distance between two states ρ^A and σ^A on the same register is denoted as $\|\rho^A - \sigma^A\|_1$, where

$$\|O^A\|_1 \quad := \quad \mathrm{Tr}\left((O^\dagger O)^{\frac{1}{2}}\right) \tag{5}$$

is the trace norm for operators on system A. We sometimes omit the superscript if the system is clear from context. In operational terms, the trace distance between the two states ρ^A and σ^A is four times the best possible bias with which we can distinguish between the two states, given a single unknown copy of one of the two.

Bures distance $\mathfrak{h}$ is a fidelity based distance measure, defined for $\rho, \sigma \in \mathcal{D}(A)$ as

$$\mathfrak{h}(\rho,\sigma) \quad := \quad \sqrt{1 - \mathrm{F}(\rho,\sigma)} \ , \tag{6}$$

where fidelity F is defined as $\mathrm{F}(\rho,\sigma) := \|\sqrt{\rho}\sqrt{\sigma}\|_1$. It is the quantum analogue of Hellinger distance, which plays an important role in classical communication and information theory (see, e.g., the cut-and-paste lemma in Ref. [3]).

The following lemma, a direct consequence of the Uhlmann theorem, is called the local transition lemma [15], especially when expressed in terms of other metrics.

▶ **Lemma 3.** *Let $\rho_1,\ \rho_2 \in \mathcal{D}(A)$ have purifications $\rho_1^{AR_1}$, $\rho_2^{AR_2}$, with $|R_1| \leq |R_2|$. Then, there exists an isometry $V^{R_1\to R_2}$ such that*

$$\mathfrak{h}\left(\rho_1^A, \rho_2^A\right) \quad = \quad \mathfrak{h}\Big(V\left(\rho_1^{AR_1}\right), \rho_2^{AR_2}\Big) \ . \tag{7}$$

Bures distance is related to trace distance through a generalization of the Fuchs-van de Graaf inequalities [10]: for any $\rho_1,\ \rho_2 \in \mathcal{D}(A)$, it holds that

$$\mathfrak{h}^2(\rho_1,\rho_2) \quad \leq \quad \frac{1}{2}\,\|\rho_1-\rho_2\|_1 \quad \leq \quad \sqrt{2}\,\mathfrak{h}(\rho_1,\rho_2) \ . \tag{8}$$

In order to quantify the information content of a quantum state, we use a basic measure, von Neumann entropy, defined as

$$\mathrm{H}(A)_\rho \quad := \quad -\operatorname{Tr}\left(\rho\log\rho\right)$$

for any state $\rho \in \mathcal{D}(A)$. Here, we follow the convention that $0\log 0 = 0$, which is justified by a continuity argument. The logarithm is in base 2.

For a state $\rho^{ABC} \in \mathcal{D}(ABC)$, the mutual information between registers A, B is defined as

$$\mathrm{I}(A\!:\!B)_\rho \quad := \quad \mathrm{H}(A)+\mathrm{H}(B)-\mathrm{H}(AB) \ ,$$

and the conditional mutual information between them, given C, as

$$\mathrm{I}(A\!:\!B\,|\,C)_\rho \quad := \quad \mathrm{I}(A\!:\!BC)-\mathrm{I}(A\!:\!C) \ .$$

The following lemma, known as the Average Encoding Theorem [15, 13], formalizes the intuition that if a classical and a quantum register are weakly correlated, then they are nearly independent.

▶ **Lemma 4.** *For any $\rho^{XA} = \sum_x p_X(x)\cdot|x\rangle\langle x|^X \otimes \rho_x^A$ with a classical system X and states $\rho_x \in \mathcal{D}(A)$,*

$$\sum_x p_X(x)\cdot\mathfrak{h}^2\left(\rho_x^A,\rho^A\right) \quad \leq \quad \mathrm{I}(X\!:\!A)_\rho \ . \tag{9}$$

2.3 Quantum Information Complexity

We rely on the notion of quantum information cost of a two-party communication protocol introduced by Touchette [25]. We follow the notation associated with a two-party quantum communication protocol introduced in Section 2.1, and restrict ourselves to protocols with classical inputs XY distributed as ν.

Quantum information cost is defined in terms of the purifying register R, but is independent of the choice of purification. Given the asymmetric nature of the Augmented Index function, we consider the quantum information cost of messages from Alice to Bob and the ones from Bob to Alice separately. Such an asymmetric notion of quantum information cost was previously considered in Refs. [14, 17].

▶ **Definition 5.** Given a quantum protocol Π with classical inputs distributed as ν, the *quantum information cost* (of the messages) from Alice to Bob is defined as

$$\mathrm{QIC}_{\mathrm{A}\to\mathrm{B}}(\Pi,\nu) \quad = \quad \sum_{i \text{ odd}} \mathrm{I}(R{:}C_i \,|\, B_i) \ , \tag{10}$$

and the *quantum information cost* (of the messages) from Bob to Alice is defined as

$$\mathrm{QIC}_{\mathrm{B}\to\mathrm{A}}(\Pi,\nu) \quad = \quad \sum_{i \text{ even}} \mathrm{I}(R{:}C_i \,|\, A_i) \ . \tag{11}$$

It is immediate that quantum information cost is bounded above by quantum communication.

▶ Remark. For any quantum protocol Π with classical inputs distributed as ν, the following holds:

$$\mathrm{QIC}_{\mathrm{A}\to\mathrm{B}}(\Pi,\nu) \quad \leq \quad 2\,\mathrm{QCC}_{A\to B}(\Pi) \ , \tag{12}$$
$$\mathrm{QIC}_{\mathrm{B}\to\mathrm{A}}(\Pi,\nu) \quad \leq \quad 2\,\mathrm{QCC}_{B\to A}(\Pi) \ . \tag{13}$$

As a result, we may bound quantum communication complexity of a protocol from below by analysing its information cost.

We further restrict ourselves to "safe protocols", in which the registers $A_{\mathrm{in}}, B_{\mathrm{in}}$ are only used as control registers in the local isometries. This restriction does not affect the results in this article, for the following reason. Let Π be any protocol with classical inputs distributed as ν, in which the two parties may apply arbitrary isometries to their quantum registers. In particular, these registers include $A_{\mathrm{in}}, B_{\mathrm{in}}$ which are initialized to the input. Let Π' be the protocol with the same registers as Π and two additional quantum registers $A'_{\mathrm{in}}, B'_{\mathrm{in}}$ of the same sizes as $A_{\mathrm{in}}, B_{\mathrm{in}}$, respectively. In the protocol Π', the two parties each make a coherent local copy of their inputs into $A'_{\mathrm{in}}, B'_{\mathrm{in}}$, respectively, at the outset. The registers $A'_{\mathrm{in}}, B'_{\mathrm{in}}$ are never touched hereafter, and the two parties simulate the original protocol Π on the remaining registers. Laurière and Touchette [17] show that the quantum information cost of Π is at least as much as that of the protocol Π':

$$\mathrm{QIC}_{\mathrm{A}\to\mathrm{B}}(\Pi',\nu) \quad \leq \quad \mathrm{QIC}_{\mathrm{A}\to\mathrm{B}}(\Pi,\nu) \ , \qquad \text{and}$$
$$\mathrm{QIC}_{\mathrm{B}\to\mathrm{A}}(\Pi',\nu) \quad \leq \quad \mathrm{QIC}_{\mathrm{B}\to\mathrm{A}}(\Pi,\nu) \ .$$

Thus, the quantum information cost trade-off we show for safe protocols holds for arbitrary protocols as well.

2.4 Quantum Streaming Algorithms

We refer the reader to the text [22] for an introduction to classical streaming algorithms. Quantum streaming algorithms are similarly defined, with restricted access to the input, and with limited workspace.

In more detail, an input $x \in \Sigma^n$, where Σ is some alphabet, arrives as a *data stream*, i.e., letter by letter in the order $x_1, x_2, \ldots, x_n$. An algorithm is said to make a *pass* on the input, when it reads the data stream once in this order, processing it as described next. For an integer $T \geq 1$, a T-pass (unidirectional) *quantum streaming algorithm* $\mathcal{A}$ with space $s(n)$ and time $t(n)$ is a collection of quantum channels $\{\mathcal{A}_{i\sigma} : i \in [T], \sigma \in \Sigma\}$. Each operator $\mathcal{A}_{i\sigma}$ is a channel defined on a register of $s(n)$-qubits, and can be implemented by a uniform family of circuits of size at most $t(n)$. On input stream $x \in \Sigma^n$,

1. The algorithm starts with a register W of $s(n)$ qubits, all initialized to a fixed state, say $|0\rangle$.
2. $\mathcal{A}$ performs T sequential passes, $i = 1, \ldots, T$, on x in the order $x_1, x_2, \ldots, x_n$.
3. In the ith pass, when symbol σ is read, channel $\mathcal{A}_{i\sigma}$ is applied to W.
4. The output of the algorithm is the state in a designated sub-register W_{out} of W, at the end of the T passes.

We may allow for some pre-processing before the input is read, and some post-processing at the end of the T passes, each with time complexity different from $t(n)$. As our work applies to streaming algorithms with any time complexity, we do not consider this refinement.

The probability of correctness of a streaming algorithm is defined in the standard way. If we wish to compute a family of Boolean functions $g_n : \Sigma^n \to \{0, 1\}$, the output register W_{out} consists of a single qubit. On input x, let $\mathcal{A}(x)$ denote the random variable corresponding to the outcome when the output register is measured in the standard basis. We say $\mathcal{A}$ computes g_n with (worst-case) error $\varepsilon \in [0, 1/2]$ if for all x, $\Pr[\mathcal{A}(x) = g_n(x)] \geq 1 - \varepsilon$.

In general, the implementation of a quantum channel used by a streaming algorithm with unitary operations involves one-time use of ancillary qubits (initialized to a fixed, known quantum state, say $|0\rangle$). These ancillary qubits are in addition to the $s(n)$-qubit register that is maintained by the algorithm. Fresh qubits may be an expensive resource in practice, for example, in NMR implementations, and one may argue that they be included in the space complexity of the algorithm. The lack of ancillary qubits severely restricts the kind of computations space-bounded algorithms can perform; see, for example, Ref. [2]. We choose the definition above so as to present the results we derive in the strongest possible model. Thus, the results also apply to implementations in which the "flying qubits" needed for implementing non-unitary quantum channels are relatively easy to prepare.

In the same vein, we may provide a quantum streaming algorithm arbitrary read-only access to a sequence of random bits. In other words, we may also provide the algorithm with a register S of size at most $t(n)$ initialized to random bits from some distribution. Each quantum channel $\mathcal{A}_{i\sigma}$ now operates on registers SW, while using S only as a control register. The bounds we prove hold in this model as well.

3 Reduction from Augmented Index to DYCK(2)

The connection between low-information protocols for Augmented Index and streaming algorithms for Dyck(2) contains two steps. The first is a reduction from an intermediate multi-party communication problem Ascension, and the second is the relationship of the latter with Augmented Index.

3.1 Reduction from Ascension to DYCK(2)

In this section, we describe the connection between multi-party quantum communication protocols for the problem Ascension(m, n), and quantum streaming algorithms for Dyck(2). The reduction is an immediate generalization of the one in the classical case discovered by Magniez, Mathieu, and Nayak [19], which also works with appropriate modifications for multi-pass classical streaming algorithms [6, 12]. For the sake of completeness, we describe the reduction below.

Multi-party quantum communication protocols involving point-to-point communication may be defined as in the two-party case. As it is straightforward, and detracts from the thrust of this section, we omit a formal definition.

Let m, n be positive integers. The $(2m)$-party communication problem $\textsc{Ascension}(m, n)$ consists of computing the logical OR of m independent instances of f_n, the Augmented Index function. Suppose we denote the $2m$ parties by $\mathsf{A}_1, \mathsf{A}_2, \ldots, \mathsf{A}_m$ and $\mathsf{B}_1, \mathsf{B}_2, \ldots, \mathsf{B}_m$. Player A_i is given $x^i \in \{0,1\}^n$, player B_i is given $k^i \in [n]$, a bit z^i, and the prefix $x^i[1, k^i - 1]$ of x^i. Let $\mathbf{x} = (x^1, x^2, \ldots, x^m)$, $\mathbf{k} = (k^1, k^2, \ldots, k^m)$, and $\mathbf{z} = (z^1, z^2, \ldots, z^m)$. The goal of the communication protocol is to compute

$$F_{m,n}(\mathbf{x}, \mathbf{k}, \mathbf{z}) \quad = \quad \bigvee_{i=1}^{m} f_n(x^i, k^i, z^i) \quad = \quad \bigvee_{i=1}^{m} (x^i[k^i] \oplus z^i) \ ,$$

which is 0 if $x^i[k^i] = z^i$ for all i, and 1 otherwise.

The communication between the $2m$ parties is required to be T sequential iterations of communication in the following order, for some $T \geq 1$:

$$\begin{aligned} &\mathsf{A}_1 \to \mathsf{B}_1 \to \mathsf{A}_2 \to \mathsf{B}_2 \to \cdots \mathsf{A}_m \to \mathsf{B}_m \\ &\quad \to \mathsf{A}_m \to \mathsf{A}_{m-1} \to \cdots \to \mathsf{A}_2 \to \mathsf{A}_1 \ . \end{aligned} \tag{14}$$

In other words, for $t = 1, 2, \ldots, T$,

- for i from 1 to $m-1$, player A_i sends register $C_{\mathsf{A}_i,t}$ to B_i, then B_i sends register $C_{\mathsf{B}_i,t}$ to A_{i+1},
- A_m sends register $C_{\mathsf{A}_m,t}$ to B_m, then B_m sends register $C_{B_m,t}$ to A_m,
- for i from m down to 2, A_i sends register $C'_{\mathsf{A}_i,t}$ to A_{i-1}.

At the end of the T iterations, A_1 computes the output.

There is a bijection between instances of $\textsc{Ascension}(m, n)$ and a subset of instances of $\textsc{Dyck}(2)$ that we describe next. For any string $z = z_1 \cdots z_n \in \{a, b\}^n$, let $\overline{z}$ denote the matching string $\overline{z_n}\,\overline{z_{n-1}} \cdots \overline{z_1}$ corresponding to z. Let $z[i, j]$ denote the substring $z_i z_{i+1} \cdots z_j$ if $1 \leq i \leq j \leq n$, and the empty string ε otherwise. We abbreviate $z[i, i]$ as $z[i]$ if $1 \leq i \leq n$. Consider strings of the form

$$w \quad = \quad (x^1\,\overline{y^1}\,\overline{z^1}\,z^1\,y^1)\,(x^2\,\overline{y^2}\,\overline{z^2}\,z^2\,y^2) \cdots \ (x^m\,\overline{y^m}\,\overline{z^m}\,z^m\,y^m)\,\overline{x^m} \ \cdots \ \overline{x^2}\ \overline{x^1} \ , \tag{15}$$

where for every i, $x^i \in \{a, b\}^n$, and y^i is a suffix of x^i, i.e., $y^i = x^i[n - k^i + 2, n]$ for some $k^i \in \{1, 2, \ldots, n\}$, and $z^i \in \{a, b\}$. The string w is in $\textsc{Dyck}(2)$ if and only if, for every i, $z^i = x^i[n - k^i + 1]$. Note that these instances have length in the interval $[2m(n+1), 4mn]$.

The bijection between instances of $\textsc{Ascension}(m, n)$ and $\textsc{Dyck}(2)$ arises from a partition of the string w amongst the $2m$ players: player A_i is given x^i (and therefore $\overline{x^i}$), and player B_i is given y^i, z^i (and therefore $\overline{y^i}, \overline{z^i}$). See Figure 2 for a pictorial representation of the partition.

For ease of notation, the strings x^i in $\textsc{Ascension}(m, n)$ are taken to be the ones in $\textsc{Dyck}(2)$ with the bits *in reverse order*. This converts the suffixes y^i into prefixes of the same length.

As a consequence of the bijection above, any quantum streaming algorithm for $\textsc{Dyck}(2)$ results in a quantum protocol for $\textsc{Ascension}(m, n)$, as stated in the following lemma.

▶ **Lemma 6.** *For any $\varepsilon \in [0, 1/2]$, $n, m \in \mathbb{N}$, and for any ε-error (unidirectional) T-pass quantum streaming algorithm $\mathcal{A}$ for $\textsc{Dyck}(2)$ that on instances of size $N \in \Theta(mn)$ uses $s(N)$ qubits of memory, there exists an ε-error, T-round sequential $(2m)$-party quantum communication protocol for $\textsc{Ascension}(m, n)$ in which each message is of length $s(N)$. The protocol may use public randomness, but does not use pre-shared entanglement between any of the parties. Moreover, the local operations of any party are memory-less, i.e., do not require access to the qubits used in generating the previous messages sent by that party.*

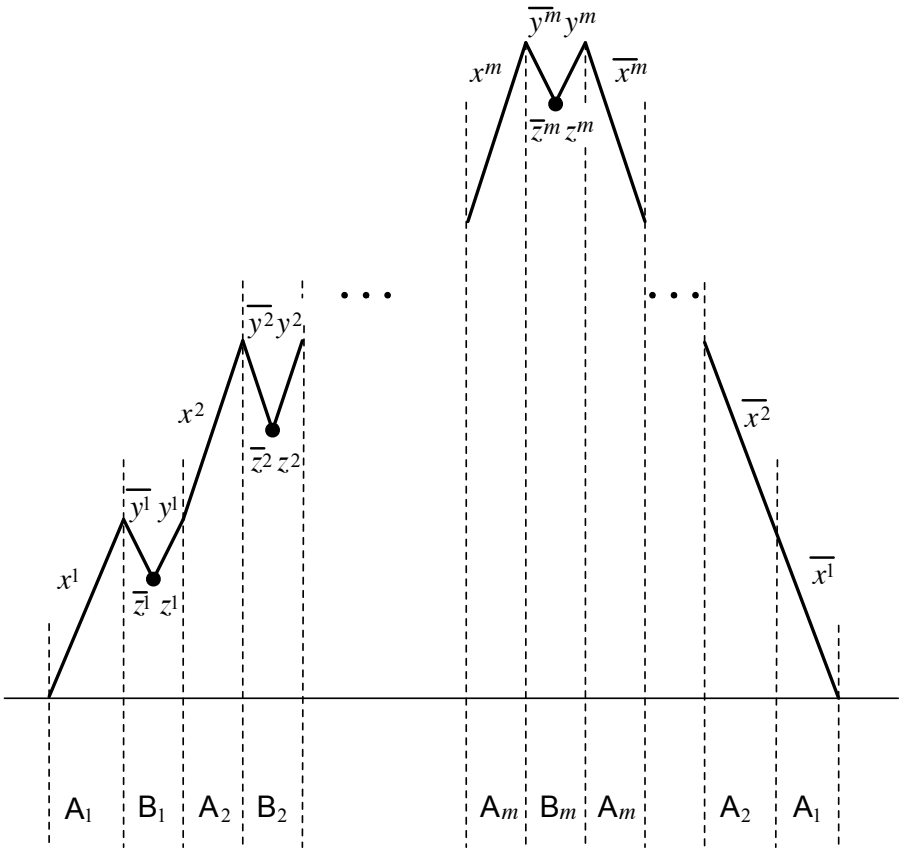


Figure 2 An instance of the form described in (15), as depicted in [19, 12]. A line segment with positive slope denotes a string over $\{a, b\}$, and a segment with negative slope denotes a string over $\{\overline{a}, \overline{b}\}$. A solid dot depicts a pair of the form $\overline{z}z$ for some $z \in \{a, b\}$. The entire string is distributed amongst $2m$ players $\mathsf{A}_1, \mathsf{B}_1, \mathsf{A}_2, \mathsf{B}_2, \ldots, \mathsf{A}_m, \mathsf{B}_m$ in a communication protocol for Ascension(m, n) as shown.

Proof. Any random sequence of bits used by the streaming algorithm is provided as shared randomness to all the $2m$ parties in the communication protocol for Ascension(m, n). Each input for the communication problem corresponds to an instance of Dyck(2), as described above. In each of the T iterations, a player simulates the quantum streaming algorithm on appropriate part of the input for Dyck(2), and sends the length $s(N)$ workspace to the next player in the sequence. (If needed, non-unitary quantum operations may be replaced with an isometry, as follows from the Stinespring Representation theorem [26].) The output of the protocol is the output of the algorithm, and is contained in the register held by the final party A_1. ◀

3.2 Reduction from Augmented Index to Ascension

Recall that Ascension(m, n) is composed of m instances of Augmented Index on n bits. Magniez, Mathieu, and Nayak [19] showed how we may derive a low-information classical protocol for Augmented Index f_n from one for Ascension(m, n) through a direct sum argument (see Refs. [6, 12] for the details of its working in the multi-pass case). This is

not as straightforward to execute as it might first appear; it entails deriving a sequence of protocols for Augmented Index in which the communication from Alice to Bob corresponds to messages from different parties in the original multi-party protocol. We show that the same kind of construction, suitably adapted to the notion of quantum information cost we use, also works in the quantum case.

Let μ_0 be the uniform distribution on the 0-inputs of the Augmented Index function f_n. If X is a uniformly random n-bit string, K is a uniformly random index from $[n]$ independent of X, and the random variable B is set as $B = X_K$, the joint distribution of $X, K, X[1, K-1], B$ is μ_0. We denote the random variables $K, X[1, K-1], B$ given as input to Bob by Y. Since $X_K = B$ under this distribution, we abbreviate Bob's input as $K, X[1, K]$. Let μ be the uniform distribution over all inputs. Under this distribution, the bit B is uniformly random, independent of XK, while XK are as above.

▶ **Lemma 7.** *Suppose $\varepsilon \in [0, 1/2]$, $n, m \in \mathbb{N}$ and that there is an ε-error, T-round sequential quantum protocol Π_{ASC} for $\textsc{Ascension}(m, n)$, that is memory-less, does not have pre-shared entanglement between any of the parties (but might use public randomness), and only has messages of length at most s (cf. Lemma 6). Then, there exists an ε-error, $2T$-message, two-party quantum protocol Π_{AI} for the Augmented Index function f_n that satisfies*

$$\text{QIC}_{\text{A}\to\text{B}}(\Pi_{\text{AI}}, \mu_0) \leq 2sT \ , \tag{16}$$

$$\text{QIC}_{\text{B}\to\text{A}}(\Pi_{\text{AI}}, \mu_0) \leq 2sT/m \ . \tag{17}$$

Proof. Starting from the $(2m)$-party protocol Π_{ASC} for $\textsc{Ascension}(m, n)$, we construct a protocol Π_j for f_n, for each $j \in [m]$.

Fix one such j. Suppose Alice and Bob get inputs x and y, respectively, where $y := (k, x[1, k-1], b)$. They embed these into an instance of $\textsc{Ascension}(m, n)$: they set $x_j = x$, and $y_j = y$. They sample the inputs for the remaining $m-1$ coordinates independently, according to μ_0. Let $X_i Y_i$, with $Y_i = (K_i, X_i[1, K_i])$, be registers corresponding to inputs drawn from μ_0 in coordinate i. Let R_i be a purification register for these, which we may decompose as $R_i^X R_i^Y$, denoting the standard purification of the $X_i Y_i$ registers. Let $S_{\text{A}} S_{\text{B}}$ be registers initialized to $\sum_s \sqrt{\nu_s}\,|ss\rangle$, so as to simulate the public random string $S \sim \nu$ used in the protocol Π_{ASC}.

For each $i \neq j$, we give X_i to Alice, and $(K_i, X_i[1, K_i])$ to Bob. For $i > j$, we give R_i to Bob, and for $i < j$, we give R_i to Alice. Then Alice and Bob simulate the roles of the $2m$ parties $(\mathsf{A}_i, \mathsf{B}_i)_{i\in[m]}$ in the following way for each of the T rounds in Π_{ASC}. For $t = 1, 2, \ldots, T$:

1. Alice simulates $\mathsf{A}_1 \to \mathsf{B}_1 \to \mathsf{A}_2 \to \cdots \to \mathsf{A}_j$, accessing the inputs for B_i, for $i < j$, in the register R_i. We denote the ancillary register she uses to simulate A_1's local isometry by D_{t1}, and for all other $i < j$, the ancillary registers she uses for B_i and A_{i+1} together by D_{ti}.
2. Alice transmits the message from A_j to B_j to Bob.
3. Bob simulates $\mathsf{B}_j \to \mathsf{A}_{j+1} \to \cdots \to \mathsf{B}_m$, accessing the input for A_i, for $i > j$, in the register R_i. For all i such that $j \leq i < m$ we denote the ancillary registers Bob uses for simulating B_i and A_{i+1}'s local isometry together by D_{ti}, and the ancillary register he uses for B_m by D_{tm}.
4. Bob transmits the message from B_m to A_m to Alice.
5. Alice simulates $\mathsf{A}_m \to \mathsf{A}_{m-1} \to \cdots \to \mathsf{A}_1$. We denote the ancillary registers Alice uses for simulating the local isometries of $\mathsf{A}_m, \mathsf{A}_{m-1}, \ldots, \mathsf{A}_1$ by E_t.

We let E_0 denote a dummy register initialized to a fixed state, say $|0\rangle$.

Since the inputs for Augmented Index for $i \neq j$ are distributed according to μ_0, the protocol Π_j computes Augmented Index for the instance (x, y) with error at most ε.

The quantum information cost from Alice to Bob $\mathrm{QIC}_{\mathrm{A}\to\mathrm{B}}(\Pi_j, \lambda)$ is bounded by $2sT$, for any distribution λ over the inputs, as each of her T messages has at most s qubits.

The bound on quantum information cost from Bob to Alice arises from the following direct sum result. Suppose that the inputs for the protocol Π_j for Augmented Index are drawn from the distribution μ_0. Denote these inputs by X_jY_j, with $Y_j = (K_j, X_j[1, K_j])$, and the corresponding purification register by R_j. We are interested in the quantum information cost $\mathrm{QIC}_{\mathrm{B}\to\mathrm{A}}(\Pi_j, \mu_0)$.

For $t \in [T]$, let C_t denote the tth message from Bob to Alice in the protocol Π_j. At the time Alice receives message C_t, her other registers are $X_1 \cdots X_m$, S_{A}, $R_1 \cdots R_{j-1}$, $(E_{r-1}D_{r1}D_{r2} \cdots D_{rj})_{r\in[t]}$. Note that the corresponding state ρ_t at that point on registers

$$X_1 \cdots X_m \; S_{\mathrm{A}} \; (E_{r-1}D_{r1}D_{r2} \cdots D_{rm})_{r\in[t]} \; R_1 \cdots R_m \; C_t$$

is the same for all derived protocols Π_j, as all of them simulate Π_{Asc} on the same input distribution $\mu_0^{\otimes m}$, using the above registers.

We have

$$\begin{aligned}
&\mathrm{QIC}_{\mathrm{B}\to\mathrm{A}}(\Pi_j, \mu_0) \\
&\quad = \sum_{t\in[T]} \mathrm{I}(R_j : C_t \mid X_1 \cdots X_m S_{\mathrm{A}} (E_{r-1}D_{r1}D_{r2} \cdots D_{rj})_{r\in[t]} R_1 \cdots R_{j-1})_{\rho_t} \\
&\quad \leq \sum_{t\in[T]} \mathrm{I}(R_j (D_{rj})_{r\in[t]} : C_t \mid X_1 \cdots X_m S_{\mathrm{A}} (E_{r-1}D_{r1} \cdots D_{r(j-1)})_{r\in[t]} R_1 \cdots R_{j-1})_{\rho_t} \; .
\end{aligned}$$

Using the chain rule, we get

$$\begin{aligned}
&\sum_{j\in[m]} \mathrm{QIC}_{\mathrm{B}\to\mathrm{A}}(\Pi_j, \mu_0) \\
&\quad \leq \sum_{t\in[T]} \mathrm{I}(R_1 \cdots R_m (D_{r1}D_{r2} \cdots D_{rm})_{r\in[t]} : C_t \mid X_1 \cdots X_m S_{\mathrm{A}} (E_{r-1})_{r\in[t]})_{\rho_t} \; .
\end{aligned}$$

Since each summand in the expression above is bounded by $2 \log |C_t| \leq 2s$, we have that the sum is bounded by $2sT$. It follows that there exists an index j^* such that

$$\mathrm{QIC}_{\mathrm{B}\to\mathrm{A}}(\Pi_{j^*}, \mu_0) \quad \leq \quad 2sT/m \; , \tag{18}$$

as desired. As noted before, $\mathrm{QIC}_{\mathrm{A}\to\mathrm{B}}(\Pi_{j^*}, \mu_0) \leq 2sT$. This completes the reduction. ◀

4 Key Technical Tools

In this section, we present the tools needed to analyze the quantum information cost of protocols. The proofs for the statements made here appear in the full version of this work [23].

In analyzing safe quantum protocols with classical inputs in the rest of the paper, we deviate slightly from the notation for the registers used in the definition of two-party protocols in Section 2.1. We refer to the input registers $A_{\mathrm{in}}, B_{\mathrm{in}}$ by X, Y, respectively. Since we focus on safe protocols, the registers XY are only used as control registers. We express Alice's local registers after the ith message is generated as XA_i, and the local registers of Bob by YB_i. As before, the message register is not included in any of the local registers, and is denoted by C_i.

We first generalize the Average Encoding Theorem [15], to relate the quality of approximation of any intermediate state in a two-party quantum communication protocol to its

information cost. This also allows us to analyze states arising from arbitrary superpositions over inputs in such protocols. The main technical ingredient used to derive the generalization is the Fawzi-Renner inequality [9].

▶ **Theorem 8** (Fawzi-Renner inequality). *For any tripartite quantum state ρ^{ACR}, there exists a recovery map $\mathcal{R}^{A\to AC}$ from register A to registers AC satisfying*

$$\mathrm{I}(C\!:\!R\,|\,A) \quad \geq \quad -2\cdot\log_2 \mathrm{F}(\rho^{ACR}, \mathcal{R}(\rho^{AR})) \ . \tag{19}$$

In particular, it follows that

$$\mathrm{I}(C\!:\!R\,|\,A) \quad \geq \quad \mathfrak{h}^2\big(\rho^{ACR}, \mathcal{R}(\rho^{AR})\big) \ . \tag{20}$$

Informally, the Superposition-Average Encoding Theorem states that if the incremental information contained in the messages received by a party thus far is "small", then she can approximate her part of the joint state "well", *without any assistance from the other party.*

▶ **Theorem 9** (Superposition-Average Encoding Theorem). *Given any safe quantum protocol Π with input registers XY initialized according to distribution ν, let*

$$|\rho_i\rangle \quad = \quad \sum_{x,y} \sqrt{\nu(x,y)}\,|xxyy\rangle^{XR_XYR_Y}\,|\rho_i^{xy}\rangle^{A_iB_iC_i}$$

be the state on registers $XYRA_iB_iC_i$ in round i with the register R initially purifying the registers XY, with a decomposition R_XR_Y into coherent copies of X and Y, respectively. Let $\varepsilon_i := \mathrm{I}(R\!:\!C_i\,|\,YB_i)$ for odd i, and $\varepsilon_i := \mathrm{I}(R\!:\!C_i\,|\,XA_i)$ for even i. There exist registers E_i, isometries V_i and states

$$|\theta_i\rangle \quad = \quad \sum_{x,y} \sqrt{\nu(x,y)}\,|xxyy\rangle^{XR_XYR_Y}\,|\theta_i^{y}\rangle^{B_iC_iE_i}$$

for odd i satisfying

$$\mathfrak{h}\Big(\rho_i^{RYB_iC_i}, \theta_i^{RYB_iC_i}\Big) \quad \leq \quad \sum_{p\leq i,\ p \text{ odd}} \sqrt{\varepsilon_p} \ , \qquad \textit{and}$$
$$V_i\,|y\rangle^Y \quad = \quad |y\rangle^Y \otimes |\theta_i^y\rangle^{B_iC_iE_i} \ ,$$

and states

$$|\sigma_i\rangle \quad = \quad \sum_{x,y} \sqrt{\nu(x,y)}\,|xxyy\rangle^{XR_XYR_Y}\,|\sigma_i^{x}\rangle^{A_iC_iE_i}$$

for even i satisfying

$$\mathfrak{h}\Big(\rho_i^{RXA_iC_i}, \sigma_i^{RXA_iC_i}\Big) \quad \leq \quad \sum_{p\leq i,\ p \text{ even}} \sqrt{\varepsilon_p} \ , \qquad \textit{and}$$
$$V_i\,|x\rangle^X \quad = \quad |x\rangle^X \otimes |\sigma_i^x\rangle^{A_iC_iE_i} \ .$$

The recently developed *Information Flow Lemma* due to Laurière and Touchette [17] allows us to analyze information transfer using an alternative notion of information cost (defined in Section 6 for Augmented Index). The lemma states that the total gain in (conditional) information by a party over all the messages is precisely the net (conditional) information gain at the end of the protocol.

▶ **Lemma 10** (Information Flow Lemma). *Given a protocol* Π*, an input state* ρ *with purifying register* R *with arbitrary decompositions* $R = R_a^{\mathrm{A}} R_b^{\mathrm{A}} R_c^{\mathrm{A}} = R_a^{\mathrm{B}} R_b^{\mathrm{B}} R_c^{\mathrm{B}}$*, the following hold:*

$$\begin{aligned}\sum_{i\geq 0} \mathrm{I}(R_a^{\mathrm{B}} : C_{2i+1} \,|\, R_b^{\mathrm{B}} B_{2i+1}) - \sum_{i\geq 1} \mathrm{I}(R_a^{\mathrm{B}} : C_{2i} \,|\, R_b^{\mathrm{B}} B_{2i}) \\ = \quad \mathrm{I}(R_a^{\mathrm{B}} : B_{\mathrm{out}} B' \,|\, R_b^{\mathrm{B}}) - \mathrm{I}(R_a^{\mathrm{B}} : B_{\mathrm{in}} \,|\, R_b^{\mathrm{B}}) \ , \quad \textit{and}\end{aligned}$$

$$\begin{aligned}\sum_{i\geq 0} \mathrm{I}(R_a^{\mathrm{A}} : C_{2i+2} \,|\, R_b^{\mathrm{A}} A_{2i+2}) - \sum_{i\geq 0} \mathrm{I}(R_a^{\mathrm{A}} : C_{2i+1} \,|\, R_b^{\mathrm{A}} A_{2i+1}) \\ = \quad \mathrm{I}(R_a^{\mathrm{A}} : A_{\mathrm{out}} A' \,|\, R_b^{\mathrm{A}}) - \mathrm{I}(R_a^{\mathrm{A}} : A_{\mathrm{in}} \,|\, R_b^{\mathrm{A}}) \ .\end{aligned}$$

The direct quantum analogue to the Cut-and-Paste Lemma [3] from classical communication complexity does not hold. We can nevertheless obtain the following weaker property, linking the states in a two-party protocol corresponding to any four possible pairs of inputs in a two-by-two rectangle. The result says that if the states corresponding to two inputs x, x' to Alice and a fixed input y to Bob are close up to a local unitary operation on Alice's side, and the states for two inputs y, y' to Bob and a fixed input x to Alice are close up to a local unitary operation on Bob's side, then, up to local unitary operations on Alice's and Bob's sides, the states for all pairs (x'', y'') of inputs in the rectangle $\{x, x'\} \times \{y, y'\}$ are close. The lemma is a variant of the hybrid argument developed in Refs. [13, 12], and is proven along the same lines. A similar, albeit slightly weaker statement may be derived from the corresponding lemmas in these articles. For example, Lemma IV.10 from Ref. [12], when adapted to the setting described above and combined with a triangle inequality, implies bounds similar to those in Eqs. (22) and (24) below. However, in the notation of the lemma below, the bounds so derived are both larger by the additive term $2h_{i-1}$.

▶ **Lemma 11** (Quantum Cut-and-Paste). *Given any safe quantum protocol* Π *with classical inputs, consider distinct inputs* x, x' *for Alice, and* y, y' *for Bob. Let* $|\rho_0\rangle^{A_0 B_0}$ *be the shared initial state of Alice and Bob for any pair* $(x'', y'') \in \{x, x'\} \times \{y, y'\}$ *of inputs. (The state* ρ_0 *may depend on the set* $\{x, x'\} \times \{y, y'\}$*.) Let* $|\rho_{i,x''y''}\rangle^{A_i B_i C_i}$ *be the state on registers* $A_i B_i C_i$ *after the* i*th message is sent, when the input is* (x'', y'')*. For odd* i*, let*

$$h_i \quad := \quad \mathfrak{h}\left(\rho_{i,xy}^{B_i C_i}, \rho_{i,x'y}^{B_i C_i}\right)$$

and $V_{i,x\to x'}^{A_i}$ *denote the unitary operation acting on* A_i *given by the local transition lemma (Lemma 3) such that*

$$h_i \quad = \quad \mathfrak{h}\left(V_{i,x\to x'}^{A_i} |\rho_{i,xy}\rangle \ , |\rho_{i,x'y}\rangle\right) \ .$$

For even i*, let*

$$h_i \quad := \quad \mathfrak{h}\left(\rho_{i,xy}^{A_i C_i}, \rho_{i,xy'}^{A_i C_i}\right)$$

and $V_{i,y\to y'}^{B_i}$ *denote the unitary operation acting on* B_i *given by the local transition lemma such that*

$$h_i \quad = \quad \mathfrak{h}\left(V_{i,y\to y'}^{B_i} |\rho_{i,xy}\rangle, |\rho_{i,xy'}\rangle\right) \ .$$

Define $V_{0,y\to y'}^{B_0} := \mathbb{I}^{B_0}$ *and* $h_0 := 1$. *Recall that* $B_i = B_{i-1}$ *for odd* i *and* $A_i = A_{i-1}$ *for even* i. *It holds that for odd* i,

$$\mathfrak{h}\Big(V_{i-1,y\to y'}^{B_i}\,|\rho_{i,xy}\rangle, |\rho_{i,xy'}\rangle\Big) \quad = \quad h_{i-1}\ , \tag{21}$$

$$\mathfrak{h}\Big(V_{i,x\to x'}^{A_i} V_{i-1,y\to y'}^{B_i}\,|\rho_{i,xy}\rangle, |\rho_{i,x'y'}\rangle\Big) \quad \leq \quad h_i + h_{i-1} + 2\sum_{j=1}^{i-2} h_j\ , \tag{22}$$

and for even i,

$$\mathfrak{h}\Big(V_{i-1,x\to x'}^{A_i}\,|\rho_{i,xy}\rangle, |\rho_{i,x'y}\rangle\Big) \quad = \quad h_{i-1}\ , \tag{23}$$

$$\mathfrak{h}\Big(V_{i,y\to y'}^{B_i} V_{i-1,x\to x'}^{A_i}\,|\rho_{i,xy}\rangle, |\rho_{i,x'y'}\rangle\Big) \quad \leq \quad h_i + h_{i-1} + 2\sum_{j=1}^{i-2} h_j\ . \tag{24}$$

5 QIC Lower Bound for Augmented Index

In this section, we establish a lower bound for the quantum information cost of protocols for Augmented Index. The proofs for the statements made here appear in the full version of this work [23].

5.1 Relating Alice's states to $\mathrm{QIC}_{\mathbf{B}\to\boldsymbol{A}}$

We study the quantum information cost of protocols for Augmented Index on input distribution μ_0 (the uniform distribution over $f_n^{-1}(0)$), and relate it to the distance between the states on two different inputs. We first focus on the quantum information cost from Bob to Alice, arising from the messages with even i's. We show that if this cost is low, then Alice's reduced states on different inputs for Bob are close to each other. (This high level intuition is the same as that described in Ref. [12].)

We state and prove our results for inputs with even length n; a similar result can be shown for odd n by suitably adapting the proof.

We consider the following purification of the input registers, corresponding to a particular preparation method for the K register, and to a preparation of the X register also depending on the preparation of register K. Recall that the content k of register K is uniformly distributed in $[n]$. The following registers are each initialized to uniform superpositions over the domain indicated: R_S^1 over $\{0,1\}$ (with a coherent copy in R_S^2), register R_J^1 over indices $j \in [n/2]$ (with a coherent copy in R_J^2), register R_L^1 over $\ell \in [n/2+1, n]$ (with a coherent copy in R_L^2). Register R_K holds a coherent copy of register K, whose content k is set to the value j in R_J^1 when R_S^1 is 0, and to ℓ when R_S^1 is 1. Depending on the value ℓ of R_L^1, the following registers are initialized to uniform superpositions to prepare the X register, itself uniform over $\{0,1\}^n$: register R_Z^1 over $z \in \{0,1\}^\ell$, and register R_W^1 over $w \in \{0,1\}^{n-\ell}$. The register X is set to $x = zw$, so together $R_Z^1 R_W^1$ hold a coherent copy of X, and a second coherent copy is held in $R_Z^2 R_W^2$. If ℓ is clear from the context, we sometimes use the notation Z and W to refer to the parts of the X register holding z and w, respectively. Depending on the value j of R_J^1, we also refer to a further decomposition $z = z'z''$ with $z' \in \{0,1\}^j$ and $z'' \in \{0,1\}^{\ell-j}$. We denote by X_{1K} the register held by Bob and containing the first $k-1$ bits of x and the verification bit b, always equal to x_k under μ_0 (X_{1K} thus contains the first k bits of x in this case); it is set to z' when R_S^1 is 0, to z when R_S^1 is 1, and register $R_{X_{1K}}$ holds a coherent copy of it.

In summary, the resulting input state $\rho_{\mu_0}^{XKX_{1K}}$ distributed according to μ_0 is purified by register R, which decomposes as

$$R \quad := \quad R_J^1 R_J^2 R_L^1 R_L^2 R_Z^1 R_Z^2 R_W^1 R_W^2 R_S^1 R_S^2 R_K R_{X_{1K}} \ .$$

Using the normalization factor $c := 1/\sqrt{(n/2)\cdot(n/2)\cdot 2^{\ell}\cdot 2^{n-\ell}\cdot 2}$, the purified state is:

$$\begin{aligned} &|\rho_0\rangle^{RXKX_{1K}} \\ &= c \sum_{j,\ell,z,w} |jj\ell\ell zzww\rangle \left(|00\rangle\,|jz'\rangle\,|zw\rangle^X\,|jz'\rangle^{KX_{1K}} + |11\rangle\,|\ell z\rangle\,|zw\rangle^X\,|\ell z\rangle^{KX_{1K}} \right) \ . \end{aligned} \tag{25}$$

Starting with the above purification and using pre-shared entanglement $|\psi\rangle^{T_{\mathrm{A}}T_{\mathrm{B}}}$ in the initial state, the state ρ_i after round i in the protocol is

$$\begin{aligned} &|\rho_i\rangle^{RXKX_{1K}A_iB_iC_i} \\ &:= c \sum_{j,\ell,z,w} |jj\ell\ell zzww\rangle \left(|00\rangle\,|jz'\rangle\,|zw\rangle\,|jz'\rangle \left|\rho_i^{zw,(j,z')}\right\rangle + |11\rangle\,|\ell z\rangle\,|zw\rangle\,|\ell z\rangle \left|\rho_i^{zw,(\ell,z)}\right\rangle \right) \ , \end{aligned} \tag{26}$$

where $\left|\rho_i^{x,(k,x[1,k])}\right\rangle$ denotes the pure state in registers $A_iB_iC_i$ conditional on joint input $(x,(k,x[1,k]))$.

Define $R_{\mathrm{A}} := R_J^1 R_L^1 R_S^1 R_K R_W^1 R_W^2$. All of R_{A}'s sub-registers except $R_W^1 R_W^2$ are classical in $\rho_i^{R_A X A_i C_i}$, since one of their coherent copies is traced out from the global purification register R. The Z part of the X register is also classical. We can write the reduced state of ρ_i on registers $R_A X A_i C_i$ as

$$\begin{aligned} &\rho_i^{R_A X A_i C_i} \\ &= c' \sum_{j,\ell,z} |j\ell\rangle\langle j\ell| \otimes \left(|0j\rangle\langle 0j| \otimes |z\rangle\langle z|^Z \otimes \rho_{i,\ell z j z'} + |1\ell\rangle\langle 1\ell| \otimes |z\rangle\langle z|^Z \otimes \rho_{i,\ell z \ell z} \right) \ , \end{aligned}$$

in which we used normalization $c' := 1/((n/2)\cdot(n/2)\cdot 2^{\ell}\cdot 2)$ and the shorthands

$$\rho_{i,\ell z k x[1,k]} \quad := \quad \mathrm{Tr}_{B_i}\left(\left|\rho_i^{\ell z k x[1,k]}\right\rangle \left\langle \rho_i^{\ell z k x[1,k]}\right| \right) \ , \qquad \text{where} \tag{27}$$

$$\left|\rho_i^{\ell z k x[1,k]}\right\rangle \quad := \quad 1/\sqrt{2^{n-\ell}} \sum_w |www\rangle^{R_W^1 R_W^2 W} \left|\rho_i^{zw,(k,x[1,k])}\right\rangle^{A_iB_iC_i} \ . \tag{28}$$

The indices $\ell z k x[1,k]$ have the following meaning: ℓ and z indicate that Alice's input register X is in superposition after the length ℓ prefix $z = x[1,\ell]$, and k and $x[1,k]$ tell us the index k in Bob's input, the prefix $x[1,k-1]$ of x given as input to Bob, and Bob's verification bit b (which is equal to x_k under μ_0), respectively. Using this notation along with the superposition-average encoding theorem, we show the following result.

▶ **Lemma 12.** *Given any even $n \geq 2$, let J and L be random variables uniformly distributed in $[n/2]$ and $[n]\setminus[n/2]$, respectively. Conditional on some value ℓ for L, let Z be a random variable chosen uniformly at random in $\{0,1\}^{\ell}$. The following then holds for any M-message safe quantum protocol Π for Augmented Index f_n, for any even $i \leq M$:*

$$\mathrm{QIC}_{\mathrm{B}\to A}(\Pi,\mu_0) \quad \geq \quad \frac{1}{2M}\,\mathbb{E}_{j\ell z\sim JLZ}\left[\mathfrak{h}^2\left(\rho_{i,\ell z j z'}^{R_W^1 R_W^2 W A_i C_i}, \rho_{i,\ell z \ell z}^{R_W^1 R_W^2 W A_i C_i} \right) \right] \ .$$

5.2 Relating Bob's states to $\mathrm{QIC}_{\mathrm{A}\to\mathrm{B}}$

We continue with the notation from the previous section, and now focus on the quantum information cost from Alice to Bob, arising from messages with odd i's. We go via an alternative notion of information cost used by Jain and Nayak [12], and studied further by Laurière and Touchette [17]. This notion is similar to the internal information cost of classical protocols (see, e.g., Refs. [3, 4]), and is called the Holevo information cost in Ref. [17].

▶ **Definition 13.** Given a safe quantum protocol Π with classical inputs, and distribution ν over inputs, the *Holevo information cost* (of the messages) from Alice to Bob in round i is defined as

$$\widetilde{\mathrm{QIC}}^{i}_{\mathrm{A}\to\mathrm{B}}(\Pi,\nu) \quad = \quad \mathrm{I}(X : B_i C_i \,|\, Y) \ ,$$

and the cumulative *Holevo information cost* (of the messages) from Alice to Bob is defined as

$$\widetilde{\mathrm{QIC}}_{\mathrm{A}\to\mathrm{B}}(\Pi,\nu) \quad = \quad \sum_{i \text{ odd}} \widetilde{\mathrm{QIC}}^{i}_{\mathrm{A}\to\mathrm{B}}(\Pi,\nu) \ . \tag{29}$$

Given a bit string z of length at least $\ell \geq 1$, let $z^{(\ell)}$ denote the string in which z_ℓ has been flipped. The following result can be inferred from the proof of Lemma 4.9 in Ref. [12].

▶ **Lemma 14.** *Given any even $n \geq 2$, let J and L be random variables uniformly distributed in $[n/2]$ and $[n] \setminus [n/2]$, respectively. Conditional on some value ℓ for L, let Z be a random variable chosen uniformly at random in $\{0,1\}^\ell$. The following holds for any M-message safe quantum protocol Π for the Augmented Index function f_n, for any odd $i \leq M$:*

$$\frac{1}{n}\,\widetilde{\mathrm{QIC}}^{i}_{\mathrm{A}\to\mathrm{B}}(\Pi,\mu_0) \quad \geq \quad \frac{1}{16}\,\mathbb{E}_{j\ell z\sim JLZ}\left[\mathfrak{h}^2\left(\rho^{B_iC_i}_{i,\ell z j z'} \ , \ \rho^{B_iC_i}_{i,\ell z^{(\ell)} j z'}\right)\right] \ ,$$

with $\rho_{i,\ell z j z'}$ defined by Eqs. (27) and (28).

For completeness, we provide in the full version of this work [23] a proof of this lemma using our notation.

Laurière and Touchette [17] prove that Holevo information cost is a lower bound on quantum information cost QIC.

▶ **Lemma 15.** *Given any M-message quantum protocol Π and any input distribution ν, the following holds for any odd $i \leq M$:*

$$\widetilde{\mathrm{QIC}}^{i}_{\mathrm{A}\to\mathrm{B}}(\Pi,\nu) \quad \leq \quad \mathrm{QIC}_{\mathrm{A}\to\mathrm{B}}(\Pi,\nu) \ .$$

This may be derived from the Information Flow Lemma (Lemma 10) by initializing the purification register R so that R^{B}_a is be a coherent copy of X and R^{B}_b is a coherent copy of Y, and R^{B}_c is a coherent copy of both X, Y.

5.3 Lower bound on QIC

By appropriately combining the above lemmas with the quantum cut-and-paste lemma, we prove a slightly weaker variant of our main lower bound on the quantum information cost of Augmented Index, i.e., Theorem 17.

▶ **Theorem 16.** *Given any even n, the following holds for any M-message safe quantum protocol Π computing the Augmented Index function f_n with error at most ε on any input:*

$$\frac{1}{4}(1-2\varepsilon) \quad \leq \quad \left(\frac{2(M+1)^2}{n}\cdot \mathrm{QIC}_{\mathrm{A}\to\mathrm{B}}(\Pi,\mu_0)\right)^{1/2} + \left(\frac{M^3}{4}\cdot \mathrm{QIC}_{\mathrm{B}\to\mathrm{A}}(\Pi,\mu_0)\right)^{1/2} \ . \tag{30}$$

The stronger version stated in Section 6 is proven similarly using a strengthening of Lemma 12.

6 A Stronger QIC Trade-off for Augmented Index

We consider a different notion of quantum information cost, more specialized to the Augmented Index function, for which we obtain better dependence on M for the information lower bound, from M^3 to M. We also show that this notion is at least $1/M$ times $\mathrm{QIC}_{\mathrm{B}\to\mathrm{A}}$, and thus we get an overall improvement by a factor of M for the M-pass streaming lower bound. The following is a precise statement of Theorem 2. Proofs for this section can be found in the full version of this work [23].

▶ **Theorem 17.** *Given any even n, the following holds for any M-message quantum protocol Π computing the Augmented Index function f_n with error ε on any input:*

$$\frac{1}{4}(1-2\varepsilon) \quad \leq \quad \left(\frac{2(M+1)^2}{n}\cdot \mathrm{QIC}_{\mathrm{A}\to\mathrm{B}}(\Pi,\mu_0)\right)^{1/2} + \left(\frac{M^2}{2}\cdot \mathrm{QIC}_{\mathrm{B}\to\mathrm{A}}(\Pi,\mu_0)\right)^{1/2} . \tag{31}$$

Our lower bound on quantum streaming algorithms for Dyck(2), Theorem 1, follows by combining this with Lemmas 6 and 7, and taking $m = n$ so that $N \in \Theta(n^2)$.

We consider the same purification of the input registers as in Section 5.1, and the following alternative notion of quantum information cost.

▶ **Definition 18.** Given a safe quantum protocol Π for Augmented Index, the *superposed-Holevo information cost* (of the messages) from Bob to Alice in round i is defined as

$$\widetilde{\mathrm{QIC}}^{i}_{\mathrm{B}\to\mathrm{A}}(\Pi,\mu_0) \quad := \quad \mathrm{I}(R_K R_J^1 R_S^1 : R_W^1 R_W^2 W A_i C_i \,|\, R_L^1 Z)_{\rho_i} \ ,$$

with ρ_i as defined in Eq. (26), and the cumulative *superposed-Holevo information cost* (of the messages) from Bob to Alice is defined as

$$\widetilde{\mathrm{QIC}}_{\mathrm{B}\to\mathrm{A}}(\Pi,\mu_0) \quad := \quad \sum_{i \text{ even}} \widetilde{\mathrm{QIC}}^{i}_{\mathrm{B}\to\mathrm{A}}(\Pi,\mu_0) \ . \tag{32}$$

Using the average encoding theorem, we show the following.

▶ **Lemma 19.** *Given any even $n \geq 2$, let J and L be random variables uniformly distributed in $[n/2]$ and $[n]\setminus[n/2]$, respectively. Conditional on some value ℓ for L, let Z be a random variable chosen uniformly at random from $\{0,1\}^\ell$. The following then holds for any M-message safe quantum protocol Π for the Augmented Index function f_n, for even $i \leq M$:*

$$\widetilde{\mathrm{QIC}}^{i}_{\mathrm{B}\to\mathrm{A}}(\Pi,\mu_0) \quad \geq \quad \frac{1}{4}\,\mathbb{E}_{j\ell z\sim JLZ}\left[\mathfrak{h}^2\left(\rho^{R_W^1 R_W^2 W A_i C_i}_{i,\ell z j z'}, \rho^{R_W^1 R_W^2 W A_i C_i}_{i,\ell z \ell z}\right)\right] \ ,$$

with $\rho_{i,\ell z k x[1,k]}$ defined by Eqs. (27) and (28).

Using the information flow lemma, we show that this notion of information cost is a lower bound on $\mathrm{QIC}_{\mathrm{B}\to\mathrm{A}}(\Pi,\mu_0)$:

▶ **Lemma 20.** *Given any M-message safe quantum protocol Π for Augmented Index and any even $i \leq M$, the following holds:*

$$\widetilde{\mathrm{QIC}}^{i}_{\mathrm{B}\to\mathrm{A}}(\Pi,\mu_0) \quad \leq \quad \mathrm{QIC}_{\mathrm{B}\to\mathrm{A}}(\Pi,\mu_0) \ .$$

The improved lower bound on QIC follows along the same lines as in Section 5.3, but we use Lemma 19 instead of Lemma 12.

Acknowledgments. We are grateful to Mark Braverman, Ankit Garg, Young Kun Ko and Jieming Mao for useful discussions related to the development of the superposition-average encoding theorem and quantum cut-and-paste lemma. We are grateful for helpful comments from the anonymous referees.

References

1 Andris Ambainis and Rūsiņš Freivalds. 1-way quantum finite automata: Strengths, weaknesses and generalizations. In *Proceedings of the 39th Annual IEEE Symposium on Foundations of Computer Science*, pages 332–341, 1998. doi:10.1109/SFCS.1998.743469.

2 Andris Ambainis, Ashwin Nayak, Amnon Ta-Shma, and Umesh Vazirani. Dense quantum coding and quantum finite automata. *Journal of the ACM*, 49(4):1–16, 2002. doi:10.1145/581771.581773.

3 Ziv Bar-Yossef, T. S. Jayram, Ravi Kumar, and D. Sivakumar. An information statistics approach to data stream and communication complexity. *Journal of Computer and System Sciences*, 68(4):702–732, 2004. Special issue on FOCS 2002. doi:10.1016/j.jcss.2003.11.006.

4 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.

5 Robin Blume-Kohout, Sarah Croke, and Daniel Gottesman. Streaming universal distortion-free entanglement concentration. *IEEE Transactions on Information Theory*, 60(1):334–350, 2014. doi:10.1109/TIT.2013.2292135.

6 Amit Chakrabarti, Graham Cormode, Ranganath Kondapally, and Andrew McGregor. Information cost tradeoffs for Augmented Index and streaming language recognition. *SIAM Journal on Computing*, 42(1):61–83, 2013. doi:10.1137/100816481.

7 Amit Chakrabarti and Ranganath Kondapally. Everywhere-tight information cost tradeoffs for Augmented Index. In *Proceedings of the 14th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, and the 15th International Workshop on Randomization and Computation*, APPROX'11/RANDOM'11, pages 448–459, 2011. doi:10.1007/978-3-642-22935-0_38.

8 Noam Chomsky and M. P. Schotzenberger. The algebraic theory of context-free languages. In P. Braffort and D. Hirschberg, editors, *Computer Programming and Formal Languages*, pages 118–161, 1963. doi:10.1016/S0049-237X(08)72023-8.

9 Omar Fawzi and Renato Renner. Quantum conditional mutual information and approximate Markov chains. *Communications in Mathematical Physics*, 340(2):575–611, 2015. doi:10.1007/s00220-015-2466-x.

10 Christopher A. Fuchs and Jeroen van de Graaf. Cryptographic distinguishability measures for quantum-mechanical states. *IEEE Transactions on Information Theory*, 45(4):1216–1227, 1999. doi:10.1109/18.761271.

11 Dmitry Gavinsky, Julia Kempe, Iordanis Kerenidis, Ran Raz, and Ronald de Wolf. Exponential separation for one-way quantum communication complexity, with applications to cryptography. *SIAM Journal on Computing*, 38(5):1695–1708, 2008. doi:10.1137/070706550.

12 Rahul Jain and Ashwin Nayak. The space complexity of recognizing well-parenthesized expressions in the streaming model: the index function revisited. *IEEE Transactions on Information Theory*, 60(10):6646–6668, 2014. doi:10.1109/TIT.2014.2339859.

13 Rahul Jain, Jaikumar Radhakrishnan, and Pranab Sen. A lower bound for the bounded round quantum communication complexity of Set Disjointness. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*, pages 220–229, 2003. doi:10.1109/SFCS.2003.1238196.

14 Iordanis Kerenidis, Mathieu Laurière, François Le Gall, and Mathys Rennela. Information cost of quantum communication protocols. *Quantum Information and Computation*, 16(3-4):181–196, 2016.

15 Hartmut Klauck, Ashwin Nayak, Amnon Ta-Shma, and David Zuckerman. Interaction in quantum communication. *IEEE Transactions on Information Theory*, 53(6):1970–1982, 2007. doi:10.1109/TIT.2007.896888.

16 Attila Kondacs and John Watrous. On the power of quantum finite state automata. In *Proceedings of the 38th Annual IEEE Symposium on Foundations of Computer Science*, pages 66–75, 1997. doi:10.1109/SFCS.1997.646094.

17 Mathieu Laurière and Dave Touchette. Information flow in quantum communication: The cost of forgetting classical information. In *Proceedings of the 8th Conference on Innovations in Theoretical Computer Science*, 2017.

18 François Le Gall. Exponential separation of quantum and classical online space complexity. *Theory of Computing Systems*, 45:188–202, 2009. doi:10.1007/s00224-007-9097-3.

19 Frédéric Magniez, Claire Mathieu, and Ashwin Nayak. Recognizing well-parenthesized expressions in the streaming model. *SIAM Journal on Computing*, 43(6):1880–1905, 2014. doi:10.1137/130926122.

20 Ashley Montanaro. The quantum complexity of approximating the frequency moments. *Quantum Information and Computation*, 16(13-14):1169–1190, 2016.

21 Cristopher Moore and James P. Crutchfield. Quantum automata and quantum grammars. *Theoretical Computer Science*, 237(1-2):275–306, 2000. doi:10.1016/S0304-3975(98)00191-1.

22 S. Muthukrishnan. *Data Streams: Algorithms and Applications*, volume 1, number 2 of *Foundations and Trends in Theoretical Computer Science*. Now Publishers Inc., Hanover, MA, USA, 2005. doi:10.1561/0400000002.

23 Ashwin Nayak and Dave Touchette. Augmented Index and quantum streaming algorithms for DYCK(2). Technical Report arXiv:1610.04937, arXiv.org Preprint, October 17, 2016.

24 Dave Touchette. Quantum information complexity and amortized communication. Technical Report arXiv:1404.3733, arXiv.org Preprint, April 14, 2014.

25 Dave Touchette. Quantum information complexity. In *Proceedings of the 47th Annual ACM on Symposium on Theory of Computing*, pages 317–326. ACM, 2015. doi:10.1145/2746539.2746613.

26 John Watrous. Theory of Quantum Information, 2015. Manuscript of a book, available at https://cs.uwaterloo.ca/~watrous/.

27 Mark M. Wilde. *Quantum Information Theory*. Cambridge University Press, Cambridge, UK, 2013. doi:10.1017/CBO9781139525343.

Separating Quantum Communication and Approximate Rank*

Anurag Anshu[1], Shalev Ben-David[2], Ankit Garg[3], Rahul Jain[4], Robin Kothari[5], and Troy Lee[6]

1 **Centre for Quantum Technologies, National University of Singapore, Singapore**
a0109169@u.nus.edu
2 **Massachusetts Institute of Technology, Cambridge, MA, USA**
shalev@mit.edu
3 **Microsoft Research New England, Cambridge, MA, USA**
garga@microsoft.com
4 **Centre for Quantum Technologies, National University of Singapore and MajuLab, Singapore**
rahul@comp.nus.edu.sg
5 **Center for Theoretical Physics, Massachusetts Institute of Technology, Cambridge, MA, USA**
rkothari@mit.edu
6 **SPMS, Nanyang Technological University and Centre for Quantum Technologies and MajuLab, Singapore**
troyjlee@gmail.com

Abstract

One of the best lower bound methods for the quantum communication complexity of a function H (with or without shared entanglement) is the logarithm of the approximate rank of the communication matrix of H. This measure is essentially equivalent to the approximate γ_2 norm and generalized discrepancy, and subsumes several other lower bounds. All known lower bounds on quantum communication complexity in the general unbounded-round model can be shown via the logarithm of approximate rank, and it was an open problem to give any separation at all between quantum communication complexity and the logarithm of the approximate rank.

In this work we provide the first such separation: We exhibit a total function H with quantum communication complexity almost quadratically larger than the logarithm of its approximate rank. We construct H using the communication lookup function framework of Anshu et al. (FOCS 2016) based on the cheat sheet framework of Aaronson et al. (STOC 2016). From a starting function F, this framework defines a new function $H = F_{\mathcal{G}}$. Our main technical result is a lower bound on the quantum communication complexity of $F_{\mathcal{G}}$ in terms of the discrepancy of F, which we do via quantum information theoretic arguments. We show the upper bound on the approximate rank of $F_{\mathcal{G}}$ by relating it to the Boolean circuit size of the starting function F.

1998 ACM Subject Classification F.1.2 Modes of Computation

Keywords and phrases Communication Complexity, Quantum Computing, Lower Bounds, log-rank, Quantum Information

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.24

* This work is partially supported by ARO grant number W911NF-12-1-0486, by the Singapore Ministry of Education and the National Research Foundation, also through NRF RF Award No. NRF-NRFF2013-13, and the Tier 3 Grant "Random numbers from quantum processes" MOE2012-T3-1-009.

32nd Computational Complexity Conference (CCC 2017).
Editor: Ryan O'Donnell; Article No. 24; pp. 24:1–24:33

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

1 Introduction

Communication complexity studies how much two parties Alice and Bob need to communicate in order to compute a function when each party only has partial knowledge of the input. The model of quantum communication complexity allows the players to send quantum messages back and forth, and measures the total number of qubits that need to be exchanged in order to compute the function. Communication complexity has become a fundamental area in theoretical computer science with applications to circuit complexity, data structures, streaming algorithms, property testing, and linear and semi-definite programs. Many of these applications require showing communication complexity *lower bounds*, which raises the importance of studying lower bound techniques in communication complexity.

In this paper we study lower bounds on quantum communication complexity. For a two-party function $F : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$, we denote by $\mathrm{Q}(F)$ the minimum number of qubits needed by a quantum protocol to compute F with error probability at most $1/3$.

One of the strongest lower bounds on $\mathrm{Q}(F)$ comes by viewing F as a Boolean $|\mathcal{X}| \times |\mathcal{Y}|$ matrix, known as the communication matrix, which we will also denote by F. The approximate rank of F, denoted $\mathrm{rk}_{1/3}(F)$, is the minimum rank of a matrix $\tilde{F}$ that is entrywise close to F, that is, satisfying $\ell_\infty(\tilde{F} - F) \leq 1/3$. Building on the work of Kremer [26] and Yao [47], Buhrman and de Wolf [13] showed that $\mathrm{Q}(F) = \Omega(\log \mathrm{rk}_{1/3}(F))$. Later, it was shown that approximate rank can also be used to lower bound quantum communication complexity with shared entanglement, denoted $\mathrm{Q}^*(F)$. More precisely, $\mathrm{Q}^*(F) = \Omega(\log \mathrm{rk}_{1/3}(F)) - O(\log\log(|\mathcal{X}| \cdot |\mathcal{Y}|))$ [31]. As this paper studies quantum communication complexity lower bounds, we will focus on the measure $\mathrm{Q}^*(F)$, which makes our results stronger.

The logarithm of the approximate rank dominates nearly all other lower bounds on quantum communication complexity, including the discrepancy method [26], the approximate trace norm [36, 33], the generalized discrepancy method [24, 36, 38], and the approximate γ_2 norm bound [33].[1] In fact, to the best of our knowledge, all known lower bounds for general two-way quantum communication can be obtained using approximate rank. Besides being a powerful lower bound method, approximate rank is a robust measure posessing several desirable properties such as error reduction, direct sum and strong direct product theorems [39], and an optimal lifting theorem [38, 40].

Given our current state of knowledge, it is consistent that $\mathrm{Q}^*(F) = O(\log \mathrm{rk}_{1/3}(F))$ for every function F, that is, the logarithm of the approximate rank *characterizes* quantum communication complexity. As it is widely believed that this is not the case, this state of affairs points to the limitations of our current lower bound techniques for quantum communication complexity.

In this paper, we show the first superlinear separation between quantum communication complexity and the logarithm of the approximate rank.

▶ **Theorem 1.** *There is a family of total functions* $F : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ *with* $\mathrm{Q}^*(F) = \widetilde{\Omega}\Big(\log^2 \mathrm{rk}_{1/3}(F)\Big)$.

As far as we are aware, Theorem 1 is the first superlinear separation between quantum communication complexity and the logarithm of the approximate rank even for *partial* functions, which are functions defined only on a subset of the domain $\mathcal{X} \times \mathcal{Y}$.[2]

[1] In fact, the generalized discrepancy method, logarithm of approximate γ_2 norm, and logarithm of approximate rank are all equivalent, up to constant mutliplicative factors and an additive logarithmic term.

[2] For partial functions, we require the approximate low-rank decomposition of the communication matrix

One alternative to approximate rank for showing lower bounds on quantum communication complexity is the recently introduced quantum information complexity [43]. This bound has been shown to dominate the logarithm of the approximate rank [10], and has nice properties like characterizing amortized quantum communication complexity. The quantum information complexity, however, is difficult to bound for an explicit function and has not yet been used to show a new lower bound in the general unbounded-round model of quantum communication complexity.

By analogy with the log rank conjecture, which postulates that $\mathrm{D}(F) = O(\mathrm{polylog}(\mathrm{rk}(F)))$, where $\mathrm{D}(F)$ is the deterministic communication complexity of F, it is natural to state an approximate log rank conjecture. The quantum version of the approximate log rank conjecture states $\mathrm{Q}^*(F) = O(\mathrm{polylog}(\mathrm{rk}_{1/3}(F)))$. Our results show that the exponent of the logarithm in such a statement must be at least 2. The largest gap we currently know between $\mathrm{D}(F)$ and $\log \mathrm{rk}(F)$ is also quadratic [20]. One could also consider a randomized version of the log rank conjecture, stating $\mathrm{R}(F) = O(\mathrm{polylog}(\mathrm{rk}_{1/3}(F)))$, where $\mathrm{R}(F)$ is the 1/3-bounded-error randomized communication complexity. This conjecture is actually known to imply the usual deterministic log rank conjecture [25]. The largest known gap between $\mathrm{R}(F)$ and $\log \mathrm{rk}_{1/3}(F)$ is 4th power [19].

Our separation is established using quantum information theoretic arguments to lower bound quantum communication complexity of a particular family of functions known as lookup functions, introduced in [5]. We use Boolean circuit size to upper bound the logarithm of approximate rank of lookup functions. We now provide an overview of lookup functions and our proof techniques.

1.1 Techniques

Many questions in communication complexity have analogs in the (usually simpler) model of query complexity. The query complexity quantity that is analogous to approximate rank is the approximate polynomial degree. Using the quantum adversary lower bound, Ambainis [2] gave a function f with an n versus $n^{1.32}$ separation between its approximate polynomial degree and quantum query complexity. This result is the main reason for the belief that there should also be a separation between the logarithm of approximate rank and quantum communication complexity. One way to do this would be to "lift" the quantum query lower bound for f into a quantum communication lower bound for a related communication problem by composing f with an appropriate communication gadget. While such a lifting theorem is known for the approximate polynomial degree [38, 40], it remains an open question to show a lifting theorem for quantum query complexity or the quantum adversary method. The lack of an analog of the adversary lower bound in the setting of quantum communication complexity is part of the difficulty of separating the logarithm of approximate rank and quantum communication complexity.

There has recently been a great deal of progress in showing new separations between complexity measures in query complexity [20, 3, 1]. The work in query complexity most closely related to ours is the *cheat sheet* method of Aaronson et al. [1]. The cheat sheet method is a way to transform a function f into its "cheat sheet" version f_{CS} so that, for some complexity measures, f_{CS} retains the hardness of f, while other complexity measures are drastically reduced by this transformation. Among other things, Aaronson et al. [1] use this

to take values between 0 and 1 even on inputs on which the function is undefined. Without this constraint it is easy to construct large partial function separations.

method to improve Ambainis' separation and give a 4th power separation between quantum query complexity and approximate polynomial degree.

[5] generalize the cheat sheet method to communication complexity. They are able to lift several query results of [1] to communication complexity, such as an example of a total function with a super-quadratic separation between its randomized and quantum communication complexities. They do this by introducing the idea of a *lookup* function. To motivate a lookup function, consider first a communication version of the familiar address function. Alice receives inputs $x \in \{0,1\}^c$ and $u_0, \ldots, u_{2^c-1} \in \{0,1\}$ and Bob receives $y \in \{0,1\}^c$ and $v_0, \ldots, v_{2^c-1} \in \{0,1\}$. The desired output is found by interpreting $x \oplus y$ as the binary representation of a number $\ell \in \{0, \ldots, 2^c - 1\}$ and outputting $u_\ell \oplus v_\ell$.

The $(F, \mathcal{G})$ lookup function $F_\mathcal{G}$ is defined by a function $F : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ and a function family $\mathcal{G} = \{G_0, \ldots, G_{2^c-1}\}$, with $G_i : (\mathcal{X}^c \times \{0,1\}^m) \times (\mathcal{Y}^c \times \{0,1\}^m) \to \{0,1\}$. Alice receives input $\mathbf{x} = (x_1, \ldots, x_c) \in \mathcal{X}^c$ and $u_0, \ldots, u_{2^c-1} \in \{0,1\}^m$ and Bob receives inputs $\mathbf{y} = (y_1, \ldots, y_c) \in \mathcal{Y}^c$ and $v_0, \ldots, v_{2^c-1} \in \{0,1\}^m$. Now the address is determined by interpreting $(F(x_1, y_1), \ldots, F(x_c, y_c)) \in \{0,1\}^c$ as an integer $\ell \in \{0, \ldots, 2^c - 1\}$ and the goal of the players is to output $G_\ell((\mathbf{x}, u_\ell), (\mathbf{y}, v_\ell))$. Note that, in contrast to the case with the address function, in a lookup function, G_ℓ can depend on $\mathbf{x}$ and $\mathbf{y}$. This is the source of difficulty in showing lower bounds for lookup functions, and also key to their interesting properties.

Lower bound. The main result of [5] showed that, given some mild restrictions on the family of functions $\mathcal{G}$, the randomized communication complexity of $F_\mathcal{G}$ is at least that of F. Our main result shows that, given mild restrictions on the function family $\mathcal{G}$, if there is a quantum protocol with q qubits of communication for $F_\mathcal{G}$, then there is a q qubit protocol for F with *non-negligible bias*. Because of the round-by-round nature of our quantum information theoretic argument, the success probability of the quantum protocol for F decays with the number of rounds of the quantum protocol for $F_\mathcal{G}$. Thus to apply this theorem, we need to start with a function F that has high quantum communication complexity even for protocols with small bias. As the discrepancy method lower bounds quantum communication complexity even with small bias, we can informally state our main theorem as follows.

▶ **Theorem 2** (Informal restatement of Corollary 29). *For any (F, G) lookup function $F_\mathcal{G}$, provided $\mathcal{G}$ satisfies certain mild technical conditions,* $\mathrm{Q}^*(F_\mathcal{G}) = \Omega(\log(1/\operatorname{disc}(F)))$.

Let us call such theorems, where we lower bound the complexity of a lookup function $F_\mathcal{G}$ (or a cheat sheet function f_{CS}) in terms of a measure of the original function F (or f), "cheat sheet theorems." Essentially optimal cheat sheet theorems have been shown in a number of computational models such as deterministic, randomized, and quantum query complexity [1] and randomized communication complexity [5]. Cheat sheet theorems are in spirit similar to joint computation results such as direct sum and direct product theorems [7, 9, 11, 16, 30, 39, 43].[3] Direct sum and direct product theorems are widely applicable tools and are often an important goal by themselves. Cheat sheet theorems have become useful tools recently and for example, the cheat sheet theorems proven in [1] were later used in [4]. We hope that our quantum cheat sheet theorem will find further applications.

[3] One point of difference is that in direct sum and direct product theorems, the lower bounds on the amount of resources (query, communication, etc.) usually scale with c, the number of copies of the function F. In the cheat sheet theorem we prove (and also in prior works), the lower bounds do not scale with c. This is due to the fact that the value of c is usually small in our applications.

We now provide a high-level overview of the proof of our quantum cheat sheet theorem. We would like to rule out the existence of a quantum protocol Π that solves the lookup function $F_{\mathcal{G}}$ and whose communication cost is much smaller than the quantum communication complexity of F (with inverse polynomial bias, for technical reasons explained below). Since Π has small communication cost, during the course of the protocol Alice and Bob do not know the value of the index $\ell = (F(x_1, y_1), \ldots, F(x_c, y_c))$. Also since there are too many cells in the array, which has length $2^c \gg \mathrm{Q}^*(F)$, and Π has small communication cost, Alice and Bob cannot talk about too many cells of the array. We first show that these two conditions imply that Alice and Bob have little information about the contents of the correct cell of the other player's array, i.e., Alice has little information about v_ℓ and Bob has little information about u_ℓ.

In the hypothesis of the theorem, we assume that G_ℓ satisfies a *nontriviality condition*: this states that $G_\ell(\mathbf{x}, \mathbf{y}, u_\ell, v_\ell)$ takes both values 0 and 1 as (u_ℓ, v_ℓ) range over all possible values. Thus the fact that Alice has little information about v_ℓ and Bob has little information about u_ℓ sounds like we have reached a contradiction already. The issue is that we do not have any control over the *bias* of G_ℓ. This situation is reminiscent of the quantum information theoretic arguments in the proof of quantum communication complexity lower bounds for the disjointness function [23]. In that case, one has to argue that a quantum protocol that solves the AND function on 2 bits exchanges non-trivial amount of information even on distributions which are extremely biased towards the AND being 0. We use similar arguments (namely the *quantum cut-and-paste* argument) to obtain a contradiction for our lookup function. Quantum cut-and-paste arguments usually have a round dependence (which is provably needed for the disjointness lower bound) but which may not be needed for our lookup function. Improving our quantum cheat sheet theorem or proving that it is tight remains an excellent open question.

At a high level our proof follows the same strategy as the proof for randomized communication complexity in [5], but the implementation of the steps of the argument is different due to the quantum nature of the protocol. A quantum communication protocol presents several challenges, such as the fact that there is no notion of a communication transcript, since it is not possible to store all the quantum messages exchanged during the protocol. Hence arguments that applied to the overall communication transcript do not work in the quantum setting. Several technical lemmas, such as the Markov chain property of classical communication protocols used in [5], fail to hold in the quantum setting.

Upper bound. We devise a general technique for proving upper bounds on the logarithm of approximate rank of lookup functions for carefully constructed function families $\mathcal{G}$. Given a circuit $\mathcal{C}$ for F, a cell in the array tries to certify the computation of F by the circuit $\mathcal{C}$. More formally, $G_\ell(\mathbf{x}, \mathbf{y}, u_\ell, v_\ell) = 1$ iff $(F(x_1, y_1), \ldots, F(x_c, y_c)) = \ell$ and $u_\ell \oplus v_\ell$ provides the values of the inputs and outputs to all the gates in $\mathcal{C}$ for each of the c different evaluations of $\mathcal{C}$ on inputs $(x_1, y_1), \ldots, (x_c, y_c)$. We show that a small circuit for F implies a good upper bound on the approximate rank of the lookup function $F_{\mathcal{G}}$.

▶ **Theorem 3** (Informal restatement of Theorem 28). *For any Boolean function F, there exists a family of functions $\mathcal{G}$ satisfying certain nontrivality conditions such that the lookup function $F_{\mathcal{G}}$ satisfies* $\log \mathrm{rk}_{1/3}(F_{\mathcal{G}}) = \widetilde{O}(\sqrt{\mathrm{size}(F)})$.

Here $\mathrm{size}(F)$ denotes the size of the smallest circuit (i.e., the one with the least number of gates) for F over some constant-sized gate set, such as the set of all 2-bit gates. The high level idea for the upper bound is the following. Suppose an all-knowing prover Merlin provided

Alice and Bob the value $\ell = (F(x_1, y_1), \ldots, F(x_c, y_c))$. Then they can "unambiguously" verify Merlin's answer with a small amount of quantum communication. Essentially they look at the ℓ^{th} cell of the array and try to find an inconsistency in the circuit values. This can then be done with quadratically less communication by a quantum protocol by using a distributed version of Grover's algorithm [21, 12]. We then show that this sort of upper bound on "unambiguously certifiable quantum communication" provides an upper bound on the log of approximate rank of the lookup function $F_{\mathcal{G}}$. A similar upper bound was also used in the query complexity separations of [1].

Putting these upper and lower bounds together, if we choose F to be the inner product function, which has exponentially small discrepancy and linear circuit size, Theorem 2 and Theorem 3 give us the desired quadratic separation between quantum communication complexty and the log of approximate rank for a lookup function $F_{\mathcal{G}}$.

One intriguing aspect of Theorem 3 is that if one can prove lower bounds on $\log \mathrm{rk}_{1/3}(F_{\mathcal{G}})$ greater than $\sqrt{n}$ for *every* nontrivial function family $\mathcal{G}$, then one proves nontrivial circuit lower bounds for F! This theorem is similar in flavor to the theorem [28, 37] that the square of the quantum query complexity of a function f is a lower bound on the formula size of f. It might seem hopeless to prove a lower bound on $\log \mathrm{rk}_{1/3}(F_{\mathcal{G}})$ for every nontrivial function family $\mathcal{G}$, but this is exactly what our quantum cheat sheet theorem achieves for quantum communication complexity, and what the results of [5] achieve for randomized communication complexity.

2 Preliminaries and notation

We will use X, Y, Z to denote random variables as well as their distributions. $x \leftarrow X$ will stand for x being sampled from the distribution of X. For joint random variables XY, Y^x will denote the distribution of $Y|X = x$.

We now state some classical complexity measures that will be used in this paper. We define quantum measures in more detail in Section 2.1 and Section 2.2. We first formally define approximate rank.

▶ **Definition 4** (Approximate rank)**.** Let $\varepsilon \in [0, 1/2)$ and F be an $|\mathcal{X}| \times |\mathcal{Y}|$ matrix. The ε-approximate rank of F is defined as

$$\mathrm{rk}_\varepsilon(F) = \min_{\tilde{F}} \left\{ \mathrm{rk}(\tilde{F}) : \forall x \in \mathcal{X}, y \in \mathcal{Y}, |\tilde{F}(x, y) - F(x, y)| \leq \varepsilon \right\}.$$

As discussed in the introduction, approximate rank lower bounds bounded-error quantum communication complexity with shared entanglement. It also lower bounds ε-error quantum communication [31]:

▶ **Fact 5.** *For any two-party function $F : \mathcal{X} \times \mathcal{Y} \to \{0, 1\}$ and $\varepsilon \in [0, 1/3]$, we have $\mathrm{Q}^*_\varepsilon(F) = \Omega(\log \mathrm{rk}_\varepsilon(F)) - O(\log \log(|\mathcal{X}| \cdot |\mathcal{Y}|))$.*

Another classical lower bound measure that we use is the discrepancy of a function [27].

▶ **Definition 6** (Discrepancy)**.** Let F be an $|\mathcal{X}| \times |\mathcal{Y}|$ Boolean-valued matrix and P a probability distribution over $\mathcal{X} \times Y$. The discrepancy of F with respect to P is

$$\mathrm{disc}_P(F) = \max_R \left| \sum_{(x,y) \in R} P(x, y)(-1)^{F(x,y)} \right| ,$$

where the maximum is taken with respect to all combinatorial rectangles R. The discrepancy of F, denoted $\mathrm{disc}(F)$, is defined as $\mathrm{disc}(F) = \min_P \mathrm{disc}_P(F)$, where the minimum is taken over all probability distributions P.

The discrepancy bound lower bounds not only bounded-error quantum communication complexity, but also quantum communication complexity with error exponentially close (in the discrepancy) to $1/2$. More precisely, we have the following [26, 33].

▶ **Theorem 7.** *Let $F : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ be a two-party function and $\varepsilon \in [0, 1/2)$. Then*

$$\mathrm{Q}^*_\varepsilon(F) = \Omega\left(\log \frac{1-2\varepsilon}{\mathrm{disc}(F)}\right).$$

Finally we define the Boolean circuit size of a function. To do this, we first fix a gate set, say the set of all gates with 2 input bits (although we could have chosen any constant instead of 2).

▶ **Definition 8** (Circuit size)**.** For a function $F : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}$, we define $\mathrm{size}(F)$ to be the size (i.e., number of gates) of the smallest circuit over the gates set of all 2-input Boolean gates that computes F.

Note that here the encoding of Alice's and Bob's input is important, since different input representations may yield different sized circuits, unlike in communication complexity. When we use this size measure, we only deal with functions defined on bits where the input encoding is clearly specified.

2.1 Quantum Information

We now introduce some quantum information theoretic notation. We assume the reader is familiar with standard notation in quantum computing [35, 45].

Let $\mathcal{H}$ be a finite-dimensional complex Euclidean space, i.e., $\mathbb{C}^n$ for some positive integer n with the usual complex inner product $\langle \cdot, \cdot \rangle$, which is defined as $\langle u, v \rangle = \sum_{i=1}^n u_i^* v_i$. We will also refer to $\mathcal{H}$ as a Hilbert space. We will usually denote vectors in $\mathcal{H}$ using braket notation, e.g., $|\psi\rangle \in \mathcal{H}$.

The ℓ_1 norm (also called the trace norm) of an operator X on $\mathcal{H}$ is $\|X\|_1 \coloneqq \mathrm{Tr}(\sqrt{X^\dagger X})$, which is also equal to (vector) ℓ_1 norm of the vector of singular values of X.

A *quantum state* (or a *density matrix* or simply a *state*) ρ is a positive semidefinite matrix on $\mathcal{H}$ with $\mathrm{Tr}(\rho) = 1$. The state ρ is said to be a *pure state* if its rank is 1, or equivalently if $\mathrm{Tr}(\rho^2) = 1$, and otherwise it is called a *mixed state*. Let $|\psi\rangle$ be a unit vector on $\mathcal{H}$, that is $\langle\psi|\psi\rangle = 1$. With some abuse of notation, we use ψ to represent the vector $|\psi\rangle$ and also the density matrix $|\psi\rangle\langle\psi|$, associated with $|\psi\rangle$. Given a quantum state ρ on $\mathcal{H}$, the *support of* ρ, denoted $\mathrm{supp}(\rho)$ is the subspace of $\mathcal{H}$ spanned by all eigenvectors of ρ with nonzero eigenvalues.

A *quantum register* A is associated with some Hilbert space $\mathcal{H}_A$. Define $|A| \coloneqq \log \dim(\mathcal{H}_A)$. Let $\mathcal{L}(A)$ represent the set of all linear operators on $\mathcal{H}_A$. We denote by $\mathcal{D}(A)$ the set of density matrices on the Hilbert space $\mathcal{H}_A$. We use subscripts (or superscripts according to whichever is convenient) to denote the space to which a state belongs, e.g, ρ with subscript A indicates $\rho_A \in \mathcal{H}_A$. If two registers A and B are associated with the same Hilbert space, we represent this relation by $A \equiv B$. For two registers A and B, we denote the combined register as AB, which is associated with Hilbert space $\mathcal{H}_A \otimes \mathcal{H}_B$. For two quantum states $\rho \in \mathcal{D}(A)$ and $\sigma \in \mathcal{D}(B)$, $\rho \otimes \sigma \in \mathcal{D}(AB)$ represents the tensor product (or Kronecker product) of ρ and σ. The identity operator on $\mathcal{H}_A$ is denoted $\mathbb{1}_A$.

Let $\rho_{AB} \in \mathcal{D}(AB)$. We define the *partial trace with respect to* A of ρ_{AB} as

$$\rho_B \coloneqq \mathrm{Tr}_A(\rho_{AB}) \coloneqq \sum_i (\langle i| \otimes \mathbb{1}_B)\rho_{AB}(|i\rangle \otimes \mathbb{1}_B),$$

where $\{|i\rangle\}_i$ is an orthonormal basis for the Hilbert space $\mathcal{H}_A$. The state $\rho_B \in \mathcal{D}(B)$ is referred to as a *reduced density matrix* or a *marginal state*. Unless otherwise stated, a missing register from subscript in a state will represent partial trace over that register. Given a $\rho_A \in \mathcal{D}(A)$, a *purification* of ρ_A is a pure state $\rho_{AB} \in \mathcal{D}(AB)$ such that $\mathrm{Tr}_B(\rho_{AB}) = \rho_A$. Any quantum state has a purification using a register B with $|B| \leq |A|$. The purification of a state, even for a fixed B, is not unique as any unitary applied on register B alone does not change ρ_A.

An important class of states that we will consider is the *classical quantum states*. They are of the form $\rho_{AB} = \sum_a \mu(a) |a\rangle\langle a|_A \otimes \rho_B^a$, where μ is a probability distribution. In this case, ρ_A can be viewed as a probability distribution and we shall continue to use the notations that we have introduced for probability distribution, for example, $\mathbb{E}_{a \leftarrow A}$ to denote the average $\sum_a \mu(a)$.

A quantum *super-operator* (or a *quantum channel* or a *quantum operation*) $\mathcal{E} : A \to B$ is a completely positive and trace preserving (CPTP) linear map (mapping states from $\mathcal{D}(A)$ to states in $\mathcal{D}(B)$). The identity operator in Hilbert space $\mathcal{H}_A$ (and associated register A) is denoted $\mathbb{1}_A$. A *unitary* operator $\mathcal{U}_A : \mathcal{H}_A \to \mathcal{H}_A$ is such that $\mathcal{U}_A^\dagger \mathcal{U}_A = \mathcal{U}_A \mathcal{U}_A^\dagger = \mathbb{1}_A$. The set of all unitary operations on register A is denoted by $\mathcal{U}(A)$.

A 2-outcome quantum measurement is defined by a collection $\{M, \mathbb{1} - M\}$, where $0 \preceq M \preceq \mathbb{1}$ is a positive semidefinite operator, where $A \preceq B$ means $B - A$ is positive semidefinite. Given a quantum state ρ, the probability of getting outcome corresponding to M is $\mathrm{Tr}(\rho M)$ and getting outcome corresponding to $\mathbb{1} - M$ is $1 - \mathrm{Tr}(\rho M)$.

2.1.1 Distance measures for quantum states

We now define the distance measures we use and some properties of these measures. Before defining the distance measures, we introduce the concept of *fidelity* between two states, which is not a distance measure but a similarity measure.

▶ **Definition 9** (Fidelity)**.** Let $\rho_A, \sigma_A \in \mathcal{D}(A)$ be quantum states. The fidelity between ρ and σ is defined as

$$\mathrm{F}(\rho_A, \sigma_A) \coloneqq \left\| \sqrt{\rho_A}\sqrt{\sigma_A} \right\|_1 .$$

For two pure states $|\psi\rangle$ and $|\phi\rangle$, we have $\mathrm{F}(|\psi\rangle\langle\psi|, |\phi\rangle\langle\phi|) = |\langle\psi|\phi\rangle|$. We now introduce the two distance measures we use.

▶ **Definition 10** (Distance measures)**.** Let $\rho_A, \sigma_A \in \mathcal{D}(A)$ be quantum states. We define the following distance measures between these states.

$$\begin{aligned} \text{Trace distance:} \quad & \Delta(\rho_A, \sigma_A) \coloneqq \frac{1}{2}\|\rho_A - \sigma_A\|_1 \\ \text{Bures metric:} \quad & \mathrm{B}(\rho_A, \sigma_A) \coloneqq \sqrt{1 - \mathrm{F}(\rho_A, \sigma_A)}. \end{aligned}$$

Note that for any two quantum states ρ_A and σ_A, these distance measures lie in $[0, 1]$. The distance measures are 0 if and only if the states are equal, and the distance measures are 1 if and only if the states have orthogonal support, i.e., if $\rho_A \rho_B = 0$.

Conveniently, these measures are closely related.

▶ **Fact 11.** *For all quantum states $\rho_A, \sigma_A \in \mathcal{D}(A)$, we have*

$$1 - \mathrm{F}(\rho_A, \sigma_A) \leq \Delta(\rho_A, \sigma_A) \leq \sqrt{2} \cdot \mathrm{B}(\rho_A, \sigma_A).$$

Proof. The Fuchs-van de Graaf inequalities [18, 45] state that

$$1 - \mathrm{F}(\rho_A, \sigma_A) \leq \Delta(\rho_A, \sigma_A) \leq \sqrt{1 - \mathrm{F}^2(\rho_A, \sigma_A)}.$$

Our fact follows from this and the relation $1 - \mathrm{F}^2(\rho_A, \sigma_A) \leq 2 - 2\mathrm{F}(\rho_A, \sigma_A)$. ◀

A fundamental fact about quantum states is Uhlmann's theorem [44].

▶ **Fact 12** (Uhlmann's theorem). *Let $\rho_A, \sigma_A \in \mathcal{D}(A)$. Let $\rho_{AB} \in \mathcal{D}(AB)$ be a purification of ρ_A and $\sigma_{AB} \in \mathcal{D}(AB)$ be a purification of σ_A with. There exists a unitary $\mathcal{U} : \mathcal{H}_B \to \mathcal{H}_B$ such that*

$$\mathrm{F}(|\theta\rangle\langle\theta|_{AB}, |\rho\rangle\langle\rho|_{AB}) = \mathrm{F}(\rho_A, \sigma_A),$$

where $|\theta\rangle_{AB} = (\mathbb{1}_A \otimes \mathcal{U}) |\sigma\rangle_{AB}$. Trivially, the same holds for the Bures metric B *as well.*

We now review some properties of the Bures metric that we use in our proofs.

▶ **Fact 13** (Facts about B). *For all quantum states $\rho_A, \rho'_A, \sigma_A, \sigma'_A \in \mathcal{D}(A)$, we have the following.*

▶ **Fact 13.A** (Triangle inequality [14]). *The following triangle inequality and a weak triangle inequality hold for the Bures metric and the square of the Bures metric.*

1. $\mathrm{B}(\rho_A, \sigma_A) \leq \mathrm{B}(\rho_A, \tau_A) + \mathrm{B}(\tau_A, \sigma_A)$.
2. $\mathrm{B}^2(\rho_A^1, \rho_A^{t+1}) \leq t \cdot \sum_{i=1}^{t} \mathrm{B}^2(\rho_A^i, \rho_A^{i+1})$.

▶ **Fact 13.B** (Product states). $\mathrm{B}(\rho_A \otimes \sigma_A, \rho'_A \otimes \sigma'_A) \leq \mathrm{B}(\rho_A, \rho'_A) + \mathrm{B}(\sigma_A, \sigma'_A)$. *Additionally, if $\sigma_A = \sigma'_A$ then $\mathrm{B}(\rho_A \otimes \sigma_A, \rho'_A \otimes \sigma'_A) = \mathrm{B}(\rho_A, \rho'_A)$.*

▶ **Fact 13.C** (Partial measurement). *For classical-quantum states $\theta_{XB}, \theta'_{XB}$ with same probability distribution on the classical part, we have*

$$\mathrm{B}^2(\theta_{XB}, \theta'_{XB}) = \mathbb{E}_{x \leftarrow X}[\mathrm{B}^2(\theta_B^x, \theta_B'^x)].$$

Proof. These facts are proved as follows.

A. Proof of part 2 follows from triangle inequality and the fact that for positive reals $a_1, a_2, \ldots a_t$,

$$\Big(\sum_i a_i\Big)^2 = \sum_i a_i^2 + 2\sum_{i<j} a_i \cdot a_j \leq \sum_i a_i^2 + \sum_{i<j}\Big(a_i^2 + a_j^2\Big) \leq t\Big(\sum_i a_i^2\Big).$$

B. Follows easily from the triangle inequality.

C. Let $\theta_{XB} = \sum_x p(x) |x\rangle\langle x| \otimes \theta_B^x$ and $\theta'_{XB} = \sum_x p(x) |x\rangle\langle x| \otimes \theta_B'^x$. Then

$$\begin{aligned}
\mathrm{F}(\theta_{XB}, \theta'_{XB}) &= \mathrm{Tr}\left(\sqrt{\sum_x p^2(x) |x\rangle\langle x| \otimes \sqrt{\theta_B^x}\theta_B'^x\sqrt{\theta_B^x}}\right) \\
&= \mathrm{Tr}\left(\sum_x p(x) |x\rangle\langle x| \otimes \sqrt{\sqrt{\theta_B^x}\theta_B'^x\sqrt{\theta_B^x}}\right) \\
&= \sum_x p(x)\mathrm{F}(\theta_B^x, \theta_B'^x) \\
&= \mathbb{E}_{x \leftarrow X}[\mathrm{F}(\theta_B^x, \theta_B'^x)],
\end{aligned}$$

which proves the fact. ◀

Finally, an important property of both these distance measures is monotonicity under quantum operations [32, 8].

▶ **Fact 14** (Monotonicity under quantum operations). *For quantum states ρ_A, $\sigma_A \in \mathcal{D}(A)$, and a quantum operation $\mathcal{E}(\cdot) : \mathcal{L}(A) \to \mathcal{L}(B)$, it holds that*

$$\Delta(\mathcal{E}(\rho), \mathcal{E}(\sigma)) \leq \Delta(\rho_A, \sigma_A) \quad \textit{and} \quad \mathrm{B}(\mathcal{E}(\rho_A), \mathcal{E}(\sigma_A)) \leq \mathrm{B}(\rho_A, \sigma_A),$$

with equality if $\mathcal{E}$ is unitary. In particular, for bipartite states $\rho_{AB}, \sigma_{AB} \in \mathcal{D}(AB)$, it holds that

$$\Delta(\rho_{AB}, \sigma_{AB}) \geq \Delta(\rho_A, \sigma_A) \quad \textit{and} \quad \mathrm{B}(\rho_{AB}, \sigma_{AB}) \geq \mathrm{B}(\rho_A, \sigma_A).$$

2.1.2 Mutual information and relative entropy

We start with the following fundamental information theoretic quantities. We refer the reader to the excellent sources for quantum information theory [46, 45] for further study.

▶ **Definition 15.** Let $\rho_A \in \mathcal{D}(A)$ be a quantum state and $\sigma_A \in \mathcal{D}(A)$ be another quantum state on the same space with $\mathrm{supp}(\rho_A) \subset \mathrm{supp}(\sigma_A)$. We then define the following.

$$\text{von Neumann entropy:} \quad \mathrm{S}(\rho_A) \coloneqq -\mathrm{Tr}(\rho_A \log \rho_A).$$
$$\text{Relative entropy:} \quad \mathrm{S}(\rho_A \| \sigma_A) \coloneqq \mathrm{Tr}(\rho_A \log \rho_A) - \mathrm{Tr}(\rho_A \log \sigma_A).$$

We now define mutual information and conditional mutual information.

▶ **Definition 16** (Mutual information). Let $\rho_{ABC} \in \mathcal{D}(ABC)$ be a quantum state. We define the following measures.

$$\text{Mutual information:} \quad \mathbb{I}(A:B)_\rho \coloneqq \mathrm{S}(\rho_A) + \mathrm{S}(\rho_B) - \mathrm{S}(\rho_{AB}) = \mathrm{S}(\rho_{AB} \| \rho_A \otimes \rho_B)\,.$$
$$\text{Cond. mutual information:} \quad \mathbb{I}(A:B \mid C)_\rho \coloneqq \mathbb{I}(A:BC)_\rho - \mathbb{I}(A:C)_\rho.$$

We will need the following basic properties.

▶ **Fact 17** (Properties of S and $\mathbb{I}$). *Let $\rho_{ABC} \in \mathcal{D}(ABC)$ be a quantum state. We have the following.*

▶ **Fact 17.A** (Nonnegativity).

$$\mathrm{S}(A\|B)_\rho \geq 0 \textit{ and } |A| \geq \mathrm{S}(A)_\rho \geq 0$$
$$\mathbb{I}(A:B)_\rho \geq 0 \textit{ and } \mathbb{I}(A:B \mid C)_\rho \geq 0.$$

▶ **Fact 17.B** (Partial measurement). *For classical-quantum states, $\theta_{XB}, \theta'_{XB}$ with same classical distribution on register X:*

$$\mathrm{S}(\theta_{XB} \| \theta'_{XB}) = \mathbb{E}_{x \leftarrow X}[\mathrm{S}(\theta_B^x \| \theta_B'^x)].$$

▶ **Fact 17.C** (Chain rule). $\mathbb{I}(A:BC)_\rho = \mathbb{I}(A:C)_\rho + \mathbb{I}(A:B \mid C)_\rho = \mathbb{I}(A:B)_\rho + \mathbb{I}(A:C \mid B)_\rho$.

▶ **Fact 17.D** (Monotonicity). *For a quantum operation $\mathcal{E}(\cdot) : \mathcal{L}(A) \to \mathcal{L}(B)$, $\mathbb{I}(A:\mathcal{E}(B)) \leq I(A:B)$ with equality when $\mathcal{E}$ is unitary. In particular $\mathbb{I}(A:BC)_\rho \geq \mathbb{I}(A:B)_\rho$.*

▶ **Fact 17.E** (Bar hopping). $\mathbb{I}(A:BC)_\rho \geq \mathbb{I}(A:B \mid C)_\rho$, *where equality holds if* $\mathbb{I}(A:C)_\rho = 0$.

▶ **Fact 17.F** (Independence). *If $\mathbb{I}(B:C)_\rho = 0$, then $\mathbb{I}(A:BC)_\rho \geq \mathbb{I}(A:B)_\rho + \mathbb{I}(A:C)_\rho$.*

▶ **Fact 17.G** (Araki-Lieb inequality). $|\mathrm{S}(\rho_{AB}) - \mathrm{S}(\rho_B)| \leq \mathrm{S}(\rho_A)$.

▶ **Fact 17.H** (Information bound).

$$\mathbb{I}(A:BC)_\rho \leq \mathbb{I}(A:C)_\rho + 2\mathrm{S}(\rho_B).$$

▶ **Fact 17.I** (Stronger version of Pinsker's inequality). *For quantum states ρ and σ:*

$$\mathrm{S}(\rho\|\sigma) \geq 1 - \mathrm{F}(\rho,\sigma) = \mathrm{B}^2(\rho,\sigma).$$

▶ **Fact 17.J.** *For classical-quantum state (register X is classical) ρ_{XAB}:*

$$\mathbb{I}(A;B|X)_\rho = \mathbb{E}_{x\leftarrow X}\mathrm{S}(\rho^x_{AB}\|\rho^x_A \otimes \rho^x_B) \geq \mathbb{E}_{x\leftarrow X}\mathrm{B}^2\left(\rho^x_{AB}, \rho^x_A \otimes \rho^x_B\right).$$
$$\mathbb{I}(X;A) = \mathrm{S}(\rho_{XA}\|\rho_X \otimes \rho_A) = \mathbb{E}_{x\leftarrow X}\mathrm{S}(\rho^x_A\|\rho_A).$$
$$\mathbb{I}(X;A) = \mathbb{I}(f(X)X;A), \text{ where } f \text{ is any function.}$$

Proof. These facts are proved as follows.

A. For nonnegativity of relative entropy, see [35, Theorem 11.7]. For nonnegativity of mutual information and conditional mutual information, see [46, Theorem 11.6.1] and [46, Theorem 11.7.1].

B. Let $\theta_{XB} = \sum_x p(x)\,|x\rangle\langle x| \otimes \theta^x_B$ and $\theta'_{XB} = \sum_x p(x)\,|x\rangle\langle x| \otimes \theta'^x_B$. Then

$$\begin{aligned}
\mathrm{S}(\theta_{XB}\|\theta'_{XB}) &= \sum_x \mathrm{Tr}(p(x)\,|x\rangle\langle x| \otimes \theta^x_B(\log\theta_{XB} - \log\theta'_{XB})) \\
&= \sum_x p(x)\mathrm{Tr}(\theta^x_B(\log(p(x)\theta^x_B) - \log(p(x)\theta'^x_B))) \\
&= \sum_x p(x)\mathrm{Tr}(\theta^x_B(\log\theta^x_B - \log\theta'^x_B)) \\
&= \mathbb{E}_{x\leftarrow X}\mathrm{S}(\theta^x_B\|\theta'^x_B),
\end{aligned}$$

which proves the fact.

C. Follows from direct calculation.

D. See [35] [Theorem 11.15].

E. Follows from Chain rule (Fact 17.C) and Non-negativity (Fact 17.A).

F. Consider the following relations that use chain rule:

$$\begin{aligned}
\mathbb{I}(A:BC)_\rho &= \mathbb{I}(A:B)_\rho + \mathbb{I}(A:C \mid B)_\rho \\
&= \mathbb{I}(A:B)_\rho + \mathbb{I}(AB:C)_\rho - \mathbb{I}(B:C)_\rho \\
&\geq \mathbb{I}(A:B)_\rho + \mathbb{I}(A:C)_\rho.
\end{aligned}$$

The last line uses $\mathbb{I}(B:C)_\rho = 0$ and monotonicity (Fact 17.D).

G. See [35] [Section 11.3.4].

H. Consider,

$$\begin{aligned}
\mathbb{I}(A:BC)_\rho &= \mathbb{I}(A:C)_\rho + \mathbb{I}(CA:B)_\rho - \mathbb{I}(B:C) \\
&\leq \mathbb{I}(A:C)_\rho + \mathbb{I}(CA:B)_\rho \\
&\leq \mathbb{I}(A:C)_\rho + \mathrm{S}(B) + \mathrm{S}(CA) - \mathrm{S}(CAB) \\
&\leq \mathbb{I}(A:C)_\rho + 2\mathrm{S}(B). \qquad \text{(Fact 17.G)}
\end{aligned}$$

I. Using Corollary 4.2 and Proposition 4.5 in [42], we find that

$$\mathrm{S}(\rho\|\sigma) \geq -2\log \mathrm{F}(\rho,\sigma).$$

The fact now follows since for any positive $x < 1$, $2^x > 2 \cdot x^2$.

J. For the first relation, we proceed as follows, and then use Pinsker's inequality.

$$\begin{aligned}
\mathbb{I}(A:B \mid X)_\rho &= \mathbb{I}(A:BX)_\rho - \mathbb{I}(A:X)_\rho \\
&= \mathrm{S}(\rho_{ABX}\|\rho_A \otimes \rho_{BX}) - \mathrm{S}(\rho_{AX}\|\rho_A \otimes \rho_X) \\
&= \mathbb{E}_{x\leftarrow X}[\mathrm{S}(\rho^x_{AB}\|\rho_A \otimes \rho^x_B) - \mathrm{S}(\rho^x_A\|\rho_A)] \\
&= \mathbb{E}_{x\leftarrow X}[-S(\rho^x_{AB}) - \mathrm{Tr}(\rho^x_A \log \rho_A) + S(\rho^x_B) + S(\rho^x_A) + \mathrm{Tr}(\rho^x_A \log \rho_A)] \\
&= \mathbb{E}_{x\leftarrow X}[-S(\rho^x_{AB}) + S(\rho^x_B) + S(\rho^x_A)] = \mathbb{E}_{x\leftarrow X}[\mathrm{S}(\rho^x_{AB}\|\rho^x_A \otimes \rho^x_B)],
\end{aligned}$$

where in third line, we have used Fact 17.B. The second relation follows by direct calculation and Fact 13.C. The third relation follows by monotonicity under the maps $|x\rangle\langle x| \to |x\rangle\langle x| \otimes |f(x)\rangle\langle f(x)|$ and partial trace. ◀

We will need the following relation between $\mathbb{I}$ and Δ for binary classical-quantum states (see also [22]).

▶ **Claim 18.** *Let $\rho_{AB} \in \mathcal{D}(AB)$ be a classical quantum state of the form $\rho_{AB} = p\,|0\rangle\langle 0|_A \otimes \rho^0_B + (1-p)\,|1\rangle\langle 1| \otimes \rho^1_B$. Then*

$$\mathbb{I}(A:B)_\rho \leq 2\log(2) \cdot \Delta(p\rho^0_B, (1-p)\rho^1_B).$$

Proof. We drop the register index from ρ^0_B, ρ^1_B. Let $\rho_{av} = p\rho^0 + (1-p)\rho^1$. Consider

$$\begin{aligned}
\mathbb{I}(A:B)_\rho &= p\mathrm{S}\left(\rho^0\middle\|\rho_{av}\right) + (1-p)\mathrm{S}\left(\rho^1\middle\|\rho_{av}\right) \qquad \text{(Fact 17.J)} \\
&= \mathrm{S}\left(p\rho^0\middle\|\frac{1}{2}\rho_{av}\right) + \mathrm{S}\left((1-p)\rho^1\middle\|\frac{1}{2}\rho_{av}\right) - p\log(2) - (1-p)\log(2) + \mathrm{S}(p) \\
&\leq \mathrm{S}\left(p\rho^0\middle\|\frac{1}{2}\rho_{av}\right) + \mathrm{S}\left((1-p)\rho^1\middle\|\frac{1}{2}\rho_{av}\right).
\end{aligned}$$

The last inequality follows from $\mathrm{S}(p) \leq \log(2)$. Now, using [6, Theorem 9], which states that

$$\mathrm{S}\left(p\rho^0\middle\|\frac{1}{2}\rho_{av}\right) \leq \log(2)\Delta(p\rho^0,(1-p)\rho^1) \text{ and } \mathrm{S}\left((1-p)\rho^1\middle\|\frac{1}{2}\rho_{av}\right) \leq \log(2)\Delta(p\rho^0,(1-p)\rho^1),$$

the claim follows. ◀

Our next claim gives us a way to use high mutual information between two registers in a classical quantum state to make a prediction about the classical part using measurement on the quantum part.

▶ **Claim 19** (Information ⇒ prediction)**.** *Let $\rho_{AB} \in \mathcal{D}(AB)$ be a classical quantum state of the form $\rho_{AB} = p\,|0\rangle\langle 0|_A \otimes \rho^0_B + (1-p)\,|1\rangle\langle 1| \otimes \rho^1_B$. The probability of predicting A by a measurement on B is at least*

$$\frac{1}{2} + \frac{\mathbb{I}(A:B)}{2\log 2}.$$

Proof. We drop the register label B. Let M be a projector on the support of positive eigenvectors of the state $p\rho^0 - (1-p)\rho^1$. Let the measurement be $\{M, \mathbb{1} - M\}$ and first outcome imply 0 in register A and second outcome imply 1. Then probability of success is

$$\begin{aligned} p\mathrm{Tr}(\rho^0 M) + (1-p)\mathrm{Tr}(\rho^1(\mathbb{1}-M)) \\ &= (1-p) + \mathrm{Tr}((p\rho^0 - (1-p)\rho^1)M) \\ &= (1-p) + \frac{1}{2}(\|p\rho^0 - (1-p)\rho^1\|_1 + \mathrm{Tr}(p\rho^0 - (1-p)\rho^1)) \\ &= (1-p) + \frac{1}{2}(\|p\rho^0 - (1-p)\rho^1\|_1 + 2p - 1) \\ &= \frac{1}{2} + \frac{1}{2}\|p\rho^0 - (1-p)\rho^1\|_1 \\ &= \frac{1}{2} + \Delta(p\rho^0, (1-p)\rho^1). \end{aligned}$$

From Claim 18, we know that $\Delta(p\rho^0, (1-p)\rho^1) \geq \mathbb{I}(A:B)/(2\log 2)$. ◀

2.2 Quantum Communication complexity

In quantum communication complexity, two players wish to compute a classical function $F\colon \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ for some finite sets $\mathcal{X}$ and $\mathcal{Y}$. The inputs $x \in \mathcal{X}$ and $y \in \mathcal{Y}$ are given to two players Alice and Bob, and the goal is to minimize the quantum communication between them required to compute the function.

While the players have classical inputs, the players are allowed to exchange quantum messages. Depending on whether or not we allow the players arbitrary shared entanglement, we get $\mathrm{Q}(F)$, bounded-error quantum communication complexity without shared enganglement and $\mathrm{Q}^*(F)$, for the same measure with shared entanglement. Obviously $\mathrm{Q}^*(F) \leq \mathrm{Q}(F)$. In this paper we will only work with $\mathrm{Q}^*(F)$, which makes our results stronger since we prove lower bounds in this work.

Let $F\colon \mathcal{X} \times \mathcal{Y} \to \{0,1,*\}$ be a partial function, with $\mathrm{dom}(F) \coloneqq \{(x,y) \in \mathcal{X} \times \mathcal{Y} : F(x,y) \neq *\}$, and let $\varepsilon \in (0, 1/2)$.

An entanglement assisted quantum communication protocol Π for this function is as follows. Alice and Bob start with a preshared entanglement. Upon receiving inputs (x,y), where Alice gets x and Bob gets y, they exchange quantum states and then Alice applies a measurement on her qubits to output 1 or 0. Let $O(x,y)$ be the random variable output by Alice in Π, given input (x,y). Let μ be a distribution over $\mathrm{dom}(F)$.

Let inputs to Alice and Bob be given in registers X and Y in the state

$$\sum_{x,y} \mu(x,y) \left|x\right\rangle\!\left\langle x\right|_X \otimes \left|y\right\rangle\!\left\langle y\right|_Y .$$

Let these registers be purified by R_X and R_Y respectively, which are not accessible to either players. Let Alice and Bob initially hold register A_0, B_0 with shared entanglement Θ_{0,A_0B_0}. Then the initial state is

$$\left|\Psi_0\right\rangle_{XYR_XR_YA_0B_0} \coloneqq \sum_{x,y} \sqrt{\mu(x,y)}\left|xxyy\right\rangle_{XR_XYR_Y} \left|\Theta_0\right\rangle_{A_0B_0} .$$

Alice applies a unitary $U^1 : XA_0 \to XA_1C_1$ such that the unitary acts on A_0 conditioned on X. She sends C_1 to Bob. Let $B_1 \equiv B_0$ be a relabelling of Bob's register B_0. He applies $U^2 : YC_1B_1 \to YC_2B_2$ such that the unitary acts on C_1B_0 conditioned on Y. He sends C_2 to Alice. Players proceed in this fashion till end of the protocol. At any round r, let the

registers be $A_rC_rB_r$, where C_r is the message register, A_r is Alice's register and B_r is Bob's register. If r is odd, then $B_r \equiv B_{r-1}$ and if r is even, then $A_r \equiv A_{r-1}$. Let the joint state in registers $A_rC_rB_r$ be $\Theta_{r,A_rC_rB_r}$. Then the global state at round r is

$$|\Psi_r\rangle_{XYR_XR_YA_rC_rB_r} \coloneqq \sum_{x,y} \sqrt{\mu(x,y)}\, |xxyy\rangle_{XR_XYR_Y} |\Theta_r\rangle_{A_rC_rB_r}.$$

We define the following quantities.

$$\begin{aligned}
\text{Worst-case error:} \quad & \operatorname{err}(\Pi) \coloneqq \max_{(x,y)\in\operatorname{dom}(F)} \{\Pr[O(x,y) \neq F(x,y)]\}. \\
\text{Distributional error:} \quad & \operatorname{err}^{\mu}(\Pi) \coloneqq \mathbb{E}_{(x,y)\leftarrow\mu} \Pr[O(x,y) \neq F(x,y)]. \\
\text{Quantum CC of a protocol:} \quad & \operatorname{QCC}(\Pi) \coloneqq \sum_i |C_i|. \\
\text{Quantum CC of } F\text{:} \quad & \operatorname{Q}^*_{\varepsilon}(F) \coloneqq \min_{\Pi:\operatorname{err}(\Pi)\leq\varepsilon} \operatorname{QCC}(\Pi).
\end{aligned}$$

Our first fact justifies using $\varepsilon = 1/3$ by default since the exact constant does not matter since the success probability of a protocol can be boosted for QCC.

▶ **Fact 20** (Error reduction). *Let $0 < \delta < \varepsilon < 1/2$. Let Π be a protocol for F with $\operatorname{err}(\Pi) \leq \varepsilon$. There exists protocol Π' for F such that $\operatorname{err}(\Pi') \leq \delta$ and*

$$\operatorname{QCC}(\Pi') \leq O\left(\frac{\log(1/\delta)}{\left(\frac{1}{2}-\varepsilon\right)^2} \cdot \operatorname{QCC}(\Pi)\right).$$

This fact is proved by simply repeating the protocol sufficiently many times and taking the majority vote of the outputs. If the error ε is close to $1/2$, we can first reduce the error to a constant by using $O(\frac{1}{(1/2-\varepsilon)^2})$ repetitions. Then $O(\log(1/\delta))$ repetitions suffice to reduce the error down to δ. Hence the quantum communication only increases by a factor of $O\left(\frac{\log(1/\delta)}{(1/2-\varepsilon)^2}\right)$.

We have the following relation between worst-case and average-case error quantum communication complexities. It follows for example from standard application of Sion's minimax theorem [41].

▶ **Fact 21** (Minimax principle). *Let $F\colon \mathcal{X}\times\mathcal{Y} \to \{0,1,*\}$ be a partial function. Fix an error parameter $\varepsilon \in (0,1/2)$ and a quantum communication bound $q \geq 0$. Suppose $\mathcal{F}$ is a family of protocols such that for every distribution μ on $\operatorname{dom}(F)$ there exists a protocol $\Pi \in \mathcal{F}$ such that*

$$\operatorname{err}^{\mu}(\Pi) \leq \varepsilon \quad \text{and} \quad \operatorname{QCC}(\Pi) \leq q.$$

Then there exists a protocol Π' such that

$$\operatorname{err}(\Pi') \leq \varepsilon \quad \text{and} \quad \operatorname{QCC}(\Pi') \leq q.$$

Our next claim shows that having some information about the output of a Boolean function F allows us to predict the output of F with some probability greater than $1/2$.

▶ **Claim 22.** *Let $F\colon \mathcal{X}\times\mathcal{Y} \to \{0,1,*\}$ be a partial function and μ be a distribution over $\operatorname{dom}(F)$. Let XY be registers with the state $\sum_{x,y}\mu(x,y)\,|x\rangle\langle x|\otimes|y\rangle\langle y|$ and define a register F that contains the value of $F(x,y)$. Let Π be a quantum communication protocol with registers X,Y input to Alice and Bob respectively and number of rounds r (which is even). There either*

- *There exists a quantum communication protocol* Π' *for* F *with* r *rounds, with input* (X, Y) *to Alice and Bob respectively, such that*

$$\mathrm{QCC}(\Pi') = \mathrm{QCC}(\Pi) + 1, \quad \textit{and} \quad \mathrm{err}^{\mu}(\Pi') < \frac{1}{2} - \frac{\mathbb{I}(F : A_r C_r \mid X)_{\Psi_r}}{2\log(2)}.$$

- *Or, there exists a quantum communication protocol* Π' *for* F *with* r *rounds, with input* (X, Y) *to Alice and Bob respectively, such that*

$$\mathrm{QCC}(\Pi') \leq \mathrm{QCC}(\Pi), \quad \textit{and} \quad \mathrm{err}^{\mu}(\Pi') < \frac{1}{2} - \frac{\mathbb{I}(F : B_r C_r \mid Y)_{\Psi_r}}{2\log(2)}.$$

Proof. We first prove the first case. In Π', Alice and Bob run the protocol Π, after which Alice proceeds as follows. Consider the state Ψ_{r,XFA_rC_r} in registers XFA_rC_r (note that we have added a new register F to the state Ψ_r, which can be done naturally). Let

$$\Psi_{r,XFA_rC_r} = \sum_x \mu(x) \left|x\right\rangle\!\left\langle x\right|_X \otimes \Psi^x_{r,FA_rC_r}$$

be the decomposition of Ψ_{r,XFA_rC_r}, which is possible since X is classical. Note that Ψ^x_{r,FA_rC_r} is a classical quantum state between the registers F and A_rC_r. Alice, essentially applying Claim 19 makes a prediction about the content of register F. Then she outputs the prediction. Clearly,

$$\mathrm{QCC}(\Pi') = \mathrm{QCC}(\Pi) + 1\,.$$

For every input x for Alice, her prediction is successful with probability at least $1/2 + \mathbb{I}(F : A_rC_r)_{\Psi^x_r}/2\log(2)$ by Claim 19. Hence the overall success probability of Π' is at least

$$\mathbb{E}_{x \leftarrow X}\left[\frac{1}{2} + \frac{\mathbb{I}(F : A_rC_r)_{\Psi^x_r}}{2\log(2)}\right] = \frac{1}{2} + \frac{\mathbb{I}(F : A_rC_r|X)_{\Psi_r}}{2\log(2)}.$$

Second case follows with same argument, but applied on Bob' side before he sends C_r to Alice. Bob then sends the outcome to Alice instead of C_r. ◀

The following claim is used in our proof to handle the easy case of a biased input distribution.

▶ **Claim 23.** *Let* $F\colon \mathcal{X} \times \mathcal{Y} \to \{0, 1, *\}$ *be a partial function and let* μ *be a distribution over* $\mathrm{dom}(F)$*. Let* $\varepsilon \in (0, 1/2)$ *and* $c \geq 1$ *be a positive integer. For* $i \in [c]$*, let* X_i, Y_i *be registers with the state* $\sum_{x,y} \mu(x,y) \left|x\right\rangle\!\left\langle x\right|_{X_i} \otimes \left|y\right\rangle\!\left\langle y\right|_{Y_i}$ *and define register* L_i *that holds the value* $F(x_i, y_i)$*. Define* $X := X_1 \dots X_c$*,* $Y := Y_1 \dots Y_c$*, and* $L := L_1 \dots L_c$*. Let* Ψ_{XYL} *be the joint state in registers* X, Y, L*. Then either*

(a) *There exists a protocol* Π *for* F *such that* $\mathrm{QCC}(\Pi) = 1$*, and* $\mathrm{err}^{\mu}(\Pi) \leq \frac{1}{2} - \varepsilon$*, or*

(b) $\Delta(\Psi_{XL}, \Psi_X \otimes W_{L_1} \otimes \dots W_{L_c}) \leq c\varepsilon$*, where* W_{L_i} *is the maximally mixed state in register* L_i*.*

Proof. Define, $q^{x_1} := \Pr[F = 0 \mid X_1 = x_1]$. Assume $\mathbb{E}_{x_1 \leftarrow X_1}\left[\left|\frac{1}{2} - q^{x_1}\right|\right] \geq \varepsilon$. Let Π be a protocol where Alice, on input x_1, outputs 0 if $q^{x_1} \geq 1/2$ and 1 otherwise. Then,

$$\mathrm{err}^{\mu}(\Pi) = \frac{1}{2} - \mathbb{E}_{x_1 \leftarrow X_1}\left|\frac{1}{2} - q^{x_1}\right| \leq \frac{1}{2} - \varepsilon.$$

Assume otherwise $\mathbb{E}_{x_1 \leftarrow X_1}\left|\frac{1}{2} - q^{x_1}\right| < \varepsilon$. This implies

$$\Delta(\Psi_{XL}, \Psi_X \otimes W_{L_1} \otimes \dots W_{L_c}) \leq c \cdot \Delta(\Psi_{X_1L_1}, \Psi_{X_1} \otimes W_{L_1}) = c \cdot \mathbb{E}_{x_1 \leftarrow X_1}\left|\frac{1}{2} - q^{x_1}\right| < c\varepsilon,$$

where the first inequality follows from Fact 13.B. ◀

In below, let A'_r, B'_r represent Alice and Bob's registers at round r. That is, at even round r, $A'_r = A_r C_r, B'_r = B_r$ and at odd r, $A'_r = A_r, B'_r = B_r C_r$. We will need the following version of quantum-cut-and-paste lemma from [34] (also see [23] for a similar argument, where it is used to lower bound quantum communication complexity of disjointness). This is a special case of [34, Lemma 7] and we have rephrased it using our notation.

▶ **Lemma 24** (Quantum cut-and-paste). *Let Π be a quantum protocol with classical inputs and consider distinct inputs u, u' for Alice and v, v' for Bob. Let $|\Psi_{0,A_0B_0}\rangle$ be the initial shared state between Alice and Bob. Also let $\left|\Psi^{u'',v''}_{k,A'_kB'_k}\right\rangle$ be the shared state after round k of the protocol when the inputs to Alice and Bob are (u'', v'') respectively. For k odd, let*

$$h_k = \mathrm{B}\left(\Psi^{u,v}_{k,B'_k}, \Psi^{u',v}_{k,B'_k}\right)$$

and for even k, let

$$h_k = \mathrm{B}\left(\Psi^{u,v}_{k,A'_k}, \Psi^{u,v'}_{k,A'_k}\right).$$

Then

$$\mathrm{B}\left(\Psi^{u',v}_{r,A'_r}, \Psi^{u',v'}_{r,A'_r}\right) \leq h_r + h_{r-1} + 2\sum_{k=1}^{r-2} h_k.$$

The following lemma (see also [15]) formalizes the following intuition: In a quantum protocol with communication q, the amount of information that Bob has about Alice's input at any time point is at most $2q$ (note that the factor of 2 is necessary because of super-dense coding.).

▶ **Lemma 25.** *Let Π be a quantum protocol with the inputs of Alice and Bob (X, Y) being jointly distributed. Alice has an additional input U which is independent of both (X, Y). Let μ denote the distribution of inputs so that $\mu(x, u, y) = \mu(x, y)\mu(u)$. Let the total pure state after the k^{th} round of the protocol be*

$$|\Psi_k\rangle_{X\widetilde{X}Y\widetilde{Y}A'_kB'_k} = \sum_{x,y} \sqrt{\mu(x,y)\mu(u)}\, |xxuu\rangle_{X\widetilde{X}U\widetilde{U}} |yy\rangle_{Y\widetilde{Y}} |\Theta^{x,u,y}_k\rangle_{A'_kB'_k}.$$

Then

$$\mathbb{I}(B'_kY\widetilde{Y} : U|X)_{\Psi_k} \leq 2q_k.$$

Here q_k is communication cost up to round k. A similar statement holds by reversing the roles of Alice and Bob.

Proof. We prove the first inequality by induction on k. The inequality holds trivially for $k = 0$. First suppose k is even, so that Bob sent the last message. Then,

$$\begin{aligned}\mathbb{I}(B'_kY\widetilde{Y} : U|X)_{\Psi_k} &\leq \mathbb{I}(B'_{k-1}Y\widetilde{Y} : U|X)_{\Psi_{k-1}} && \text{(Fact 17.D)}\\ &\leq 2q_{k-1} \leq 2q_k,\end{aligned}$$

where the first inequality follows by induction step.

Now suppose k is odd, so that Alice sent the last message. By our notation, $B'_k \equiv C_kB_k$ where C_k is Alice's message. Then,

$$\begin{aligned}\mathbb{I}(B'_kY\widetilde{Y} : U|X)_{\Psi_k} &= \mathbb{I}(C_kB_kY\widetilde{Y} : U|X)_{\Psi_k}\\ &\leq \mathbb{I}(B_kY\widetilde{Y} : U|X)_{\Psi_k} + 2\mathrm{S}(C_k|X) && \text{(Fact 17.H)}\\ &= \mathbb{I}(B'_{k-1}Y\widetilde{Y} : U|X)_{\Psi_{k-1}} + 2\mathrm{S}(C_k|X)\\ &\leq 2q_{k-1} + 2|C_k| = 2q_k,\end{aligned}$$

where last inequality follows from induction step. ◀

3 Separation

In this section we establish the main result, a nearly quadratic separation between quantum communication complexity and the logarithm of approximate rank, which we restate below.

▶ **Theorem 1.** *There is a family of total functions* $F : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ *with* $\mathrm{Q}^*(F) = \widetilde{\Omega}\Big(\log^2 \mathrm{rk}_{1/3}(F)\Big)$.

Our proof is organized as follows. In Section 3.1 we define lookup functions, which we will use to construct the function achieving the separation in Theorem 1. Then in Section 3.2 we prove Theorem 1 using results from later sections. More precisely, we prove the upper bound on our function's approximate rank using Theorem 28, proved in Section 4. We prove the lower bound using Corollary 29, which follows from Theorem 33 in Section 5. Theorem 28 and Corollary 29 provide a black-box way of using the results of Section 4 and Section 5 without delving into their proofs.

3.1 Lookup functions

We define a simpler version of lookup functions than the ones used in [5], since we only deal with total functions in this paper. This is only for simplicity, and the lower bound shown in this paper also applies to the more general lookup functions for partial functions defined in [5].

First, for any function $F\colon \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ and integer $c > 0$, we can define a new function $F^c\colon \mathcal{X}^c \times \mathcal{Y}^c \to \{0,1\}^c$ as $F^c((x_1,\ldots,x_c),(y_1,\ldots,y_c)) = (F(x_1,y_1),\ldots,F(x_c,y_c))$, which takes c inputs to F and outputs the answers to all c inputs. F^c is simply the problem of computing F on c independent inputs and outputting all c answers.

An $(F,\mathcal{G})$-lookup function, denoted $F_{\mathcal{G}}$, is defined by a function $F\colon \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ and a family $\mathcal{G} = \{G_0,\ldots,G_{2^c-1}\}$ of functions, where each $G_i\colon (\mathcal{X}^c \times \{0,1\}^m) \times (\mathcal{Y}^c \times \{0,1\}^m) \to \{0,1\}$. It can be viewed as a generalization of the address function. Alice receives input $\mathbf{x} = (x_1,\ldots,x_c) \in \mathcal{X}^c$ and $\mathbf{u} = (u_0,\ldots,u_{2^c-1}) \in \{0,1\}^{m2^c}$ and likewise Bob receives input $\mathbf{y} = (y_1,\ldots,y_c) \in \mathcal{Y}^c$ and $\mathbf{v} = (v_0,\ldots,v_{2^c-1}) \in \{0,1\}^{m2^c}$. We refer to the inputs $(\mathbf{x},\mathbf{y})$ as the "address part" of the input and the inputs $(\mathbf{u},\mathbf{v})$ as the "array part" of the input. We will refer to u_i and v_i as a "cell" of the array. The *address*, ℓ, is determined by the evaluation of F on $(x_1,y_1),\ldots,(x_c,y_c)$, that is $\ell = F^c(\mathbf{x},\mathbf{y}) \in \{0,1\}^c$. This address (interpreted as an integer in $\{0,\ldots,2^c-1\}$) then determines which function, out of the 2^c functions G_i, the players should evaluate and which pair of cells, out of the 2^c possible pairs (u_i,v_i), of the array are relevant to the output of the function. The goal of the players is to output $G_\ell(\mathbf{x},u_\ell,\mathbf{y},v_\ell)$. The formal definition is the following.

▶ **Definition 26** ($(F,\mathcal{G})$-lookup function for total F)**.** Let $F\colon \mathcal{X}\times\mathcal{Y} \to \{0,1\}$ be a function and $\mathcal{G} = \{G_0,\ldots,G_{2^c-1}\}$ a family of functions, where each $G_i\colon (\mathcal{X}^c\times\{0,1\}^m)\times(\mathcal{Y}^c\times\{0,1\}^m) \to \{0,1\}$. An $(F,\mathcal{G})$-lookup function, denoted $F_{\mathcal{G}}$, is a function

$$F_{\mathcal{G}}\colon (\mathcal{X}^c \times \{0,1\}^{m2^c}) \times (\mathcal{Y}^c \times \{0,1\}^{m2^c}) \to \{0,1\}$$

defined as follows. Let $\mathbf{x} = (x_1,\ldots,x_c) \in \mathcal{X}^c$, $\mathbf{y} = (y_1,\ldots,y_c) \in \mathcal{Y}^c$, $\mathbf{u} = (u_0,\ldots,u_{2^c-1}) \in \{0,1\}^{m2^c}$, and $\mathbf{v} = (v_0,\ldots,v_{2^c-1}) \in \{0,1\}^{m2^c}$. Then

$$F_{\mathcal{G}}(\mathbf{x},\mathbf{u},\mathbf{y},\mathbf{v}) = G_\ell(\mathbf{x},u_\ell,\mathbf{y},v_\ell),$$

where $\ell = F^c(\mathbf{x},\mathbf{y})$.

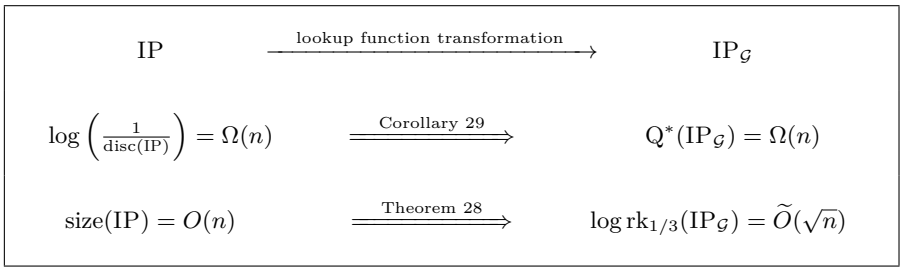


Figure 1 High-level overview of our separation. Here $\mathrm{IP} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ is the inner product function, disc is the discrepancy, and size is the circuit size.

Since we only deal with total functions F, we will not need to impose a consistency condition for instances where some input to F is outside its domain. (In [5], this condition was called "consistency outside F.")

In order to show lower bounds on the communication complexity of $F_{\mathcal{G}}$ (Theorem 33) we add two constraints on the family $\mathcal{G}$ as in [5].

▶ **Definition 27** (Nontrivial XOR family)**.** Let $\mathcal{G} = \{G_0, \ldots, G_{2^c-1}\}$ a family of communication functions, where each $G_i \colon (\mathcal{X}^c \times \{0,1\}^m) \times (\mathcal{Y}^c \times \{0,1\}^m) \to \{0,1\}$. We say that $\mathcal{G}$ is a nontrivial XOR family if the following conditions hold.

1. (Nontriviality) For all $\mathbf{x} = (x_1, \ldots, x_c) \in \mathcal{X}^c$ and $\mathbf{y} = (y_1, \ldots, y_c) \in \mathcal{Y}^c$, if we have $\ell = F^c(\mathbf{x}, \mathbf{y}) \in \{0,1\}^c$ then there exist $u, v, u', v' \in \{0,1\}^m$ such that $G_\ell(\mathbf{x}, u, \mathbf{y}, v) \neq G_\ell(\mathbf{x}, u', \mathbf{y}, v')$.
2. (XOR function) For all $i \in \{0, \ldots, 2^c - 1\}, u, u', v, v' \in \{0,1\}^m$ and $\mathbf{x} = (x_1, \ldots, x_c) \in \mathcal{X}^c, \mathbf{y} = (y_1, \ldots, y_c) \in \mathcal{Y}^c$ if $u \oplus v = u' \oplus v'$ then $G_i(\mathbf{x}, u, \mathbf{y}, v) = G_i(\mathbf{x}, u', \mathbf{y}, v')$.

The first condition simply enforces that the content of the correct part of the array, i.e., (u_ℓ, v_ℓ), is relevant to the output of the function in the sense that there is some setting of these bits that makes the function true and another setting that makes it false.

The second condition enforces that the output of the function only depends on $u_\ell \oplus v_\ell$, and not u_ℓ and v_ℓ individually. This is just one way of combining the arrays of Alice and Bob to form one virtual array that contains 2^c cells. Other combining functions are also possible.

3.2 Separation

We can now prove the separation using results from Section 4 and Section 5. Our proof strategy is depicted in Figure 1.

The separating function is going to be a lookup function $F_{\mathcal{G}}$ defined by a function $F \colon \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ and a function family $\mathcal{G} = \{G_0, \ldots, G_{2^c-1}\}$. We will choose F to be the well-known inner product function $\mathrm{IP} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ defined as

$$\mathrm{IP}(x, y) = \bigoplus_{i=1}^{n} (x_i \wedge y_i).$$

The communication complexity of the inner product function is well understood and is $\Theta(n)$ in all the models discussed in this paper. In fact, even $\log \text{sign-rank}(F) = \Theta(n)$ [17], where sign-rank(F) is defined as the minimum rank of a matrix G such that $\ell_\infty(F - G) < 1/2$.

To define our function family $\mathcal{G}$, we use the following theorem proved in Section 4.

▶ **Theorem 28.** *Let F be a total function with circuit size* $\mathrm{size}(F)$*. Then for all $c > 0$, there exists a nontrivial family of XOR functions $\mathcal{G} = \{G_0, G_1, \ldots, G_{2^c-1}\}$, such that*

$$\log \mathrm{rk}_{1/3}(F_{\mathcal{G}}) = \widetilde{O}(c^{3/2}\sqrt{\mathrm{size}(F)}).$$

This theorem gives us a function family $\mathcal{G}$ and proves that for this family we have

$$\log \mathrm{rk}_{1/3}(\mathrm{IP}_{\mathcal{G}}) = \widetilde{O}(c^{3/2}\sqrt{\mathrm{size}(\mathrm{IP})}) = \widetilde{O}(c^{3/2}\sqrt{n}), \tag{1}$$

where we use the fact that $\mathrm{size}(\mathrm{IP}) = O(n)$. This follows because IP is a parity of size n composed with an AND function on two bits, and has a circuit of size $O(n)$ consisting of a $\log n$-depth tree of fanin-2 XOR gates with fanin-2 AND gates at the bottom.

To show the lower bound, we use the following corollary of Theorem 33.

▶ **Corollary 29.** *Let $F_{\mathcal{G}}$ be an $(F, \mathcal{G})$-lookup function for a function F and a nontrivial family of XOR functions $\mathcal{G} = \{G_0, G_1, \ldots, G_{2^c-1}\}$ with $c = \Theta(\log(\mathrm{Q}^*(F)))$. Then*

$$\mathrm{Q}^*(F_{\mathcal{G}}) = \Omega(\log(1/\mathrm{disc}(F))).$$

Here $\mathrm{disc}(F)$ is the discrepancy of F (Definition 6). Since $\log(1/\mathrm{disc}(\mathrm{IP})) = \Omega(n)$ [27, Example 3.19], using Theorem 33 we have

$$\mathrm{Q}^*(\mathrm{IP}_{\mathcal{G}}) = \Omega(\log(1/\mathrm{disc}(\mathrm{IP}))) = \Omega(n). \tag{2}$$

We can now choose $c = \Theta(\log n)$ to satisfy the conditions of Corollary 29. Thus (1) yields

$$\log \mathrm{rk}_{1/3}(\mathrm{IP}_{\mathcal{G}}) = \widetilde{O}(\sqrt{n}),$$

which together with (2) gives us $\mathrm{Q}^*(\mathrm{IP}_{\mathcal{G}}) = \widetilde{\Omega}(\log^2(\mathrm{rk}_{1/3}(\mathrm{IP}_{\mathcal{G}})))$, proving Theorem 1.

4 Upper bound on approximate rank of lookup functions

The aim of this section is to prove Theorem 28. Proving this will require some work and we will need to carefully choose our function family $\mathcal{G} = \{G_0, \ldots, G_{2^c-1}\}$. To do this, we first introduce the concept of an *unambiguous* lookup function.

▶ **Definition 30.** Let $F_{\mathcal{G}}$ be an $(F, \mathcal{G})$-lookup function for a function $F : \mathcal{X} \times \mathcal{Y} \to \{0, 1\}$ and a function family $\mathcal{G} = \{G_0, G_1, \ldots, G_{2^c-1}\}$. We say that $F_{\mathcal{G}}$ is an *unambiguous* lookup function if G_ℓ evaluating to 1 certifies that $F^c(\mathbf{x}, \mathbf{y}) = \ell$. That is, for all $\mathbf{x}, u, \mathbf{y}, v$, $G_\ell(\mathbf{x}, u, \mathbf{y}, v) = 1 \Rightarrow F^c(\mathbf{x}, \mathbf{y}) = \ell$.

Note that not all lookup functions are unambiguous even if we enforce the nontrivial XOR family condition (Definition 27), since the condition for when G_i evaluates to 1 need not even depend on $\mathbf{x}$ and $\mathbf{y}$. For example, $G_i(\mathbf{x}, u, \mathbf{y}, v)$ could simply be some nonconstant function of the string $u \oplus v$. However, the condition of unambiguity is quite natural, and the lookup functions used in prior work are unambiguous lookup functions (or can be slightly modified to be unambiguous).

The advantage of unambiguous lookup functions is that we can upper bound their approximate rank as follows.

▶ **Lemma 31.** *Let $F_{\mathcal{G}}$ be an unambiguous $(F, \mathcal{G})$-lookup function. Then we have*

$$\log \mathrm{rk}_{1/3}(F_{\mathcal{G}}) = O(c \cdot \max_i \mathrm{Q}^*(G_i)).$$

Proof. We start by observing that the unambiguity condition implies that for any input $(\mathbf{x}, \mathbf{u}, \mathbf{y}, \mathbf{v})$, at most one of the functions $G_i(\mathbf{x}, u_i, \mathbf{y}, v_i)$ equals 1. Indeed, only $G_\ell(\mathbf{x}, u_\ell, \mathbf{y}, v_\ell)$ can potentially evaluate to 1, where $\ell = F^c(\mathbf{x}, \mathbf{y})$.

In other words, when $F_\mathcal{G}(\mathbf{x}, \mathbf{u}, \mathbf{y}, \mathbf{v}) = 1$ we must have $G_\ell(\mathbf{x}, u_\ell, \mathbf{y}, v_\ell) = 1$ for $\ell = F^c(x, y)$ and $G_i(\mathbf{x}, u_i, \mathbf{y}, v_i) = 0$ for all $i \neq \ell$. On the other hand, when $F_\mathcal{G}(\mathbf{x}, \mathbf{u}, \mathbf{y}, \mathbf{v}) = 0$ we must have $G_i(\mathbf{x}, u_i, \mathbf{y}, v_i) = 0$ for all $i \in \{0, \ldots, 2^c - 1\}$.

This means the communication matrix of $F_\mathcal{G}$ equals the sum of the communication matrices of G_i over all i. More precisely, we extend the definition of G_i to have it take all of $(\mathbf{x}, \mathbf{u}, \mathbf{y}, \mathbf{v})$ as input in the natural way (i.e., it ignores all the other cells of the array except u_i and v_i). This observation directly yields

$$\operatorname{rk}(F_\mathcal{G}) \leq \sum_{i=0}^{2^c-1} \operatorname{rk}(G_i).$$

The same inequality does not immediately hold for approximate rank, because the errors in the approximation can add up. So even though $A = \sum_i B_i$, if $\tilde{B}_i$ satisfies $\ell_\infty(\tilde{B}_i - B_i) \leq 1/3$, it is not necessarily the case that $\ell_\infty(A - \sum_i \tilde{B}_i) \leq 1/3$. However, if each $\tilde{B}_i$ is an excellent approximation to B_i, then their sum will still be a good approximation to A. More precisely, it is still the case that

$$\operatorname{rk}_{1/3}(F_\mathcal{G}) \leq \sum_{i=0}^{2^c-1} \operatorname{rk}_\varepsilon(G_i),$$

where $\varepsilon \leq 2^{-c}/3$, since the definition of approximate rank allows error at most $1/3$. This yields

$$\operatorname{rk}(F_\mathcal{G}) \leq 2^c \max_i \operatorname{rk}_\varepsilon(G_i) \implies \log \operatorname{rk}_{1/3}(F_\mathcal{G}) \leq c + \max_i \log \operatorname{rk}_\varepsilon(G_i).$$

Since log of approximate rank lower bounds quantum communication complexity, we have that $\log \operatorname{rk}_\varepsilon(G_i) \leq \mathrm{Q}^*_\varepsilon(G_i)$. By using standard error reduction, we have that $\mathrm{Q}^*_\varepsilon(G_i)$ for $\varepsilon = 2^{-c}/3$ is at most $O(c\,\mathrm{Q}^*(G_i))$. Hence $\log \operatorname{rk}_{1/3}(F_\mathcal{G}) = O(c \cdot \max_i \mathrm{Q}^*(G_i))$. ◀

To prove Theorem 28, we need a tool for taking a function F and finding a collection $\mathcal{G}$ such that $F_\mathcal{G}$ is an unambiguous lookup function, and $\mathrm{Q}^*(G_i)$ is small for all $G_i \in \mathcal{G}$. The following lemma provides such a tool.

▶ **Lemma 32.** *Let $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ be a total function with circuit size $\operatorname{size}(F)$ (i.e., F can be computed by a Boolean circuit with $\operatorname{size}(F)$ gates of constant fanin).*

Then for all $c > 0$, there exists a nontrivial family $\mathcal{G} = \{G_0, G_1, \ldots, G_{2^c-1}\}$ of XOR functions, such that $F_\mathcal{G}$ is an unambiguous lookup function and for all $i \in \{0, \ldots, 2^c - 1\}$,

$$\mathrm{Q}^*(G_i) = \widetilde{O}(\sqrt{c \operatorname{size}(F)}).$$

Proof. We need to construct functions $G_i(\mathbf{x}, u, \mathbf{y}, v)$ that lead to an unambiguous lookup function (Definition 30), that are a nontrivial XOR family (Definition 27) and have $\mathrm{Q}^*(G_i) = \widetilde{O}(\sqrt{c \operatorname{size}(F)})$.

Each G_i will check that $u_i \oplus v_i$ has a very special type of certificate that proves that $F^c(x, y) = i$. If it contains such a certificate, G_i will output 1 and otherwise it will output 0. This takes care of the unambiguity condition. Since G_i only depends on $u_i \oplus v_i$, it will be an XOR family and since it only evaluates to 1 on a certificate, it will be nontrivial.

We now construct the certificate. Let $\text{size}(F) = m$, which means that there is a circuit that takes in (x, y) as input and outputs $F(x, y)$ using at most m constant fanin gates. The cell $u_i \oplus v_i$ will contain c certificates, each certifying that the corresponding input to F evaluates to correct bit of i. For one instance of F, the certificate is constructed as follows. The certificate has to provide a full evaluation of the circuit of size m on (x, y) by providing the correct values for the inputs and outputs of all m gates. The final gate should, of course, evaluate the claimed output value for F. The inputs to the first level, which are inputs belonging to either Alice or Bob, should be consistent with the true inputs that Alice and Bob hold. For a circuit of size m, a certificate of this sort has size $\widetilde{O}(m)$ (with a log factor to account for describing the labels of gates), and hence the entire certificate has size $\widetilde{O}(cm)$.

If the inputs are consistent with Alice's and Bob's input, and all the gates are evaluated correctly, then the output of the circuit will be $F(x, y)$ and the output string for all c circuits will indeed be $F^c(\mathbf{x}, \mathbf{y}) = \ell$. If this output string is consistent with i, then G_i accepts and otherwise rejects.

It is easy to see that $\mathcal{G}$ satisfies the first two properties we wanted. It remains to upper bound $\mathrm{Q}^*(G_i)$. As a warmup, note that the deterministic communication complexity of G_i is at most $\widetilde{O}(cm)$. This is because Alice and Bob can simply send all of u_i and v_i to each other, which costs $\widetilde{O}(cm)$ communication. They can then check that their inputs are correct, the circuit evaluation is correct, and the circuits evaluate to i.

A similar algorithm, using Grover's algorithm to search for a discrepancy, yields the quantum algorithm. Alice and Bob first check that the $O(cm)$ inputs in the circuits (there are $O(m)$ inputs per F, and there are c copies of F) are consistent with their part of the input using $\widetilde{O}(\sqrt{cm})$ communication using Grover's algorithm. They can then Grover search over all cm gates to check if their inputs and outputs are consistent, which again takes $\widetilde{O}(\sqrt{cm})$ communication. The final step is to check that the output bits equal i. This takes $\widetilde{O}(\sqrt{c})$ communication using Grover search. Hence the total quantum communication complexity of G_i is $\widetilde{O}(\sqrt{cm}) = \widetilde{O}(\sqrt{c\ \text{size}(F)})$. ◀

Lemma 31 and Lemma 32 straightforwardly imply Theorem 28.

5 Lower bound on quantum communication complexity of lookup functions

In this section, we prove our main theorem, which is the following:

▶ **Theorem 33.** *Let $F : \mathcal{X} \times \mathcal{Y} \to \{0, 1, *\}$ be a (partial) function, $c \geq 5 \log(\mathrm{Q}^*_{1/3}(F))$ and $r \geq 1$ be an integer. Let $\mathcal{G} = \{G_0, \ldots, G_{2^c-1}\}$ be a nontrivial family of XOR functions where each $G_i : (\mathcal{X}^c \times \{0,1\}^m) \times (\mathcal{Y}^c \times \{0,1\}^m) \to \{0,1\}$, and let $F_{\mathcal{G}}$ be the $(F, \mathcal{G})$-lookup function. Let $\delta = \frac{1}{10^9 cr^2}$. For any $1/3$-error r-round protocol Π for $F_{\mathcal{G}}$, there exists a $\frac{1}{2} - \frac{\delta}{3}$-error protocol Π' for F such that*

$$\mathrm{QCC}(\Pi') = O(\mathrm{QCC}(\Pi)).$$

Before proving this, we show how it implies the corollary used in Section 3, which we restate.

▶ **Corollary 29.** *Let $F_{\mathcal{G}}$ be an $(F, \mathcal{G})$-lookup function for a function F and a nontrivial family of XOR functions $\mathcal{G} = \{G_0, G_1, \ldots, G_{2^c-1}\}$ with $c = \Theta(\log(\mathrm{Q}^*(F)))$. Then*

$$\mathrm{Q}^*(F_{\mathcal{G}}) = \Omega(\log(1/\mathrm{disc}(F))).$$

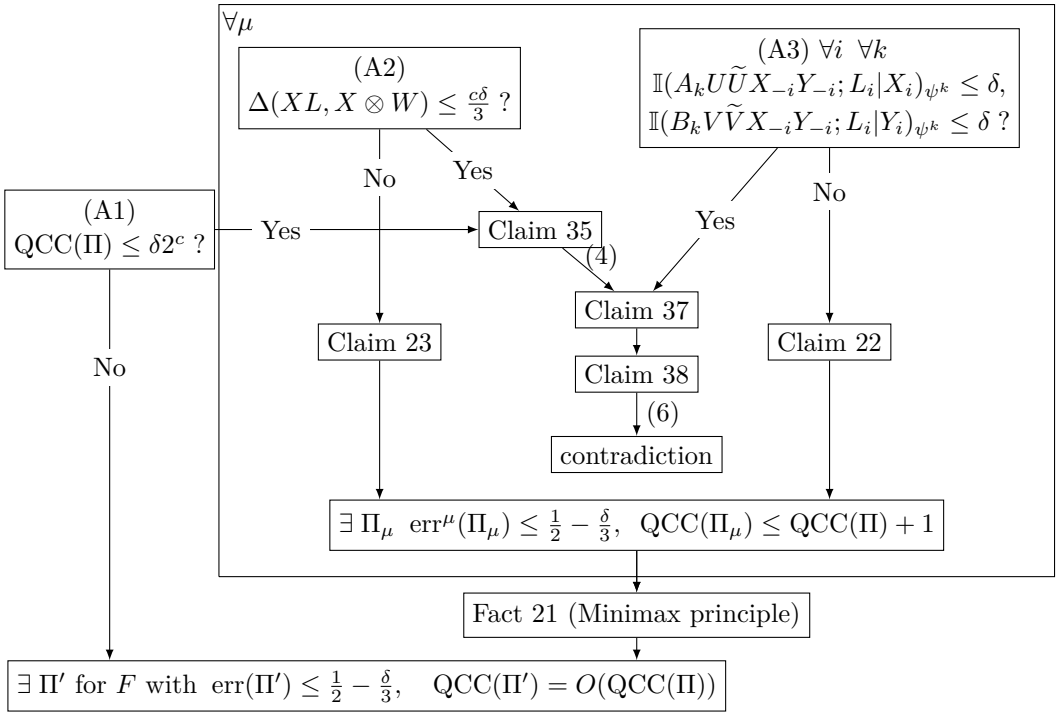


Figure 2 The structure of the proof of Theorem 33. Note that Claim 35 and Claim 37 only follow if both of their incoming arcs hold.

Proof. Let Π be a protocol for $F_{\mathcal{G}}$ with $\mathrm{QCC}(\Pi) = \mathrm{Q}^*(F)$. Then from Theorem 33, we have $\mathrm{Q}^*_\varepsilon(F) = O(\mathrm{Q}^*(F_{\mathcal{G}}))$, where $\varepsilon = \frac{1}{2} - \frac{\delta}{3}$, $\delta = \frac{1}{10^9 cr^2}$, and $r \le \mathrm{QCC}(\Pi) = \mathrm{Q}^*(F_{\mathcal{G}}))$ is the number of rounds in Π. Now from Theorem 7, we know that $\mathrm{Q}^*_\varepsilon(F) = \Omega\big(\log \frac{1-2\varepsilon}{\mathrm{disc}(F)}\big)$. Combining these with the fact that $cr^2 = O(\mathrm{Q}^*(F_{\mathcal{G}}))$ we get

$$\mathrm{Q}^*(F_{\mathcal{G}}) = \Omega\left(\log \frac{1-2\varepsilon}{\mathrm{disc}(F)}\right) = \Omega\left(\log\left(\frac{1}{\mathrm{disc}(F)}\right) - \log(cr^2)\right)$$

$$= \Omega\left(\log\left(\frac{1}{\mathrm{disc}(F)}\right) - \log \mathrm{Q}^*(F_{\mathcal{G}})\right),$$

which implies the statement to be proved, as $\mathrm{Q}^*(F_{\mathcal{G}}) = \omega(\log \mathrm{Q}^*(F_{\mathcal{G}}))$. ◀

Proof of Theorem 33. We explain here the overall structure of the argument which is also displayed visually in Figure 2.

Rule out trivial protocols. We first rule out the easy case where the protocol we are given, Π, has high quantum communication cost. More precisely, we check if the following condition holds.

$$\mathrm{QCC}(\Pi) < \delta 2^c. \tag{A1}$$

If this does not hold then $\mathrm{QCC}(\Pi) \ge \delta 2^c = \Omega(\mathrm{Q}^*(F))$. By choosing the protocol whose communication complexity is $\mathrm{Q}^*(F)$, we obtain a protocol Π' for F with $\mathrm{QCC}(\Pi') = \mathrm{Q}^*(F) = O(\mathrm{QCC}(\Pi))$ and we are done. Hence for the rest of the proof we may assume (A1).

Protocols correct on a distribution. Instead of directly constructing a protocol Π' for F that is correct on all inputs with bounded error, we instead construct for every distribution μ on $\mathrm{dom}(F)$, a protocol Π_μ that does well on μ and then use Fact 21 to construct our final protocol. More precisely, for every μ over $\mathrm{dom}(F)$ we construct a protocol Π_μ for F that has the following properties:

$$\mathrm{QCC}(\Pi_\mu) = \mathrm{QCC}(\Pi) + 1 \qquad \text{and} \qquad \mathrm{err}^\mu(\Pi_\mu) < 1/2 - \delta/3. \tag{3}$$

Hence for the remainder of the proof let μ be any distribution over $\mathrm{dom}(F)$ and our aim is to construct a protocol satisfying (3).

Construct a distribution for $\mathbf{F_\mathcal{G}}$. Using the distribution μ on $\mathrm{dom}(F)$, we now construct a distribution over the inputs to $F_\mathcal{G}$. Let the random variable T be defined as follows:

$$T := (X_1, \ldots, X_c, U_0, \ldots, U_{2^c-1}, Y_1, \ldots, Y_c, V_0, \ldots, V_{2^c-1}),$$

where for all $i \in [c]$, X_iY_i is distributed according to μ and independent of all other random variables and for $j \in \{0, \ldots, 2^c - 1\}$, U_jV_j are uniformly distributed in $\{0,1\}^{2m}$ and independent of all other variables. For $i \in [c]$, we define $L_i := F(X_i, Y_i)$. We also define $X := (X_1, \ldots, X_c)$, $Y := (Y_1, \ldots, Y_c)$, $L := (L_1, \ldots, L_c)$, $U := (U_0, \ldots, U_{2^c-1})$ and $V := (V_1, \ldots, V_{2^c-1})$. Lastly, for $i \in [c]$, we define $X_{-i} := X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_c$ and $X_{<i} := X_1, \ldots, X_{i-1}$. Similar definitions hold for L and Y. Let A_k, B_k be the registers of Alice and Bob after round k of protocol Π. The total pure state after round k can be written as follows:

$$|\psi_k\rangle_{X\widetilde{X}U\widetilde{U}Y\widetilde{Y}V\widetilde{V}A_kB_k} = \sum_{x,u,y,v} \sqrt{\mu_T(x,u,y,v)}\, |xx\rangle_{X\widetilde{X}} |uu\rangle_{U\widetilde{U}} |yy\rangle_{Y\widetilde{Y}} |vv\rangle_{V\widetilde{V}} |\psi_k^{x,u,y,v}\rangle_{A_kB_k}$$

Here μ_T is the distribution of the random variable T. $\widetilde{X}, \widetilde{U}, \widetilde{Y}, \widetilde{V}$ are registers that purify the classical inputs X, U, Y, V respectively.

Rule out easy distributions μ. We now show that if μ is such that the output of $F(X,Y)$ is predictable simply by looking at Alice's input X, then this distribution is easy and we can construct a protocol Π_μ that does well on this distribution since Alice can simply guess the value of $F(X,Y)$ after seeing X. More precisely, we check if the following condition holds.

$$\Delta(XL, X \otimes W) \leq c\delta/3, \tag{A2}$$

where W is the uniform distribution on $\{0,1\}^c$.

If the condition does not hold, we invoke Claim 23 with $\varepsilon = \delta/3$. Then we must be in case (a) of this claim and hence we get the desired protocol Π_μ. Therefore we can assume (A2) holds.

Construct new protocols $\mathbf{\Pi_i}$. We now define a collection of protocols Π_i for each $i \in [c]$. Π_i is a protocol in which Alice and Bob receive inputs from $\mathrm{dom}(F)$. We construct Π_i as follows: Given the input pair (X_i, Y_i) distributed according to μ, Alice and Bob use shared entanglement $X_{-i}\widetilde{X_{-i}}Y_{-i}\widetilde{Y_{-i}}$ (Alice holds $X_{-i}\widetilde{X_{-i}}$ and Bob holds $Y_{-i}\widetilde{Y_{-i}}$), where $X_{-i}Y_{-i}$ are distributed according to $\mu^{\otimes c-1}$ and $\widetilde{X_{-i}}\widetilde{Y_{-i}}$ purify $X_{-i}Y_{-i}$ in a canonical way. They also use shared entanglement $U\widetilde{U}V\widetilde{V}$ (Alice holds $U\widetilde{U}$ and Bob holds $V\widetilde{V}$), where U and V are uniformly distributed and $\widetilde{U}\widetilde{V}$ purify UV in a canonical way. Note that Alice and Bob now have inputs XU and YV distributed according to T. They then run protocol Π. It is clear that for all $i \in [c]$, $\mathrm{QCC}(\Pi_i) = \mathrm{QCC}(\Pi)$.

Rule out informative protocols Π_i. If any of the protocols Π_i that we constructed has a lot of information about L_i, then we can use Claim 22 to design a protocol for F. Hence, we can assume that for each $1 \leq k \leq r$,

$$\mathbb{I}(A_k U\widetilde{U} X_{-i}Y_{-i}; L_i|X_i)_{\psi_k}, \mathbb{I}(B_k V\widetilde{V} X_{-i}Y_{-i}; L_i|Y_i)_{\psi_k} \leq \delta. \tag{A3}$$

Obtain a contradiction. We have already established that (A1), (A2), and (A3) must hold, otherwise we have obtained our protocol Π_μ. We will now show that if (A1), (A2), and (A3) simultaneously hold, then we obtain a contradiction. To show this, we use some claims that are proved after this theorem.

First we apply Claim 34 to get the following from (A1) and (A2).

$$\forall k \in \{1,\ldots,r\}: \quad \mathbb{E}_{x,l\leftarrow XL}\mathrm{B}^2\left(\psi^x_{k,B_kY\widetilde{Y}V\widetilde{V}U_l}, \psi^x_{k,B_kY\widetilde{Y}V\widetilde{V}} \otimes \psi_{U_l}\right) \leq \frac{q}{2^c} + \frac{c\delta}{3}. \tag{4}$$

Here $q = \mathrm{QCC}(\Pi)/2$. Intuitively this claim asserts that for a typical x and ℓ, Bob (conditioned on $X = x$) has very little information about the cell U_ℓ at the end of round k, which is quantified by saying their joint state is close to being a product state. This would be false without assuming (A1) because if there was no upper bound on the communication in Π, then Alice could simply communicate all of U, in which case Bob would have a lot of information about any U_j. We need (A2) as well, since otherwise it is possible that the correct answer ℓ is easily predicted by Alice by looking at her input alone, in which case she can send over the contents of cell U_ℓ to Bob. A symmetric statement also follows with Alice and Bob interchanged.

We then apply Claim 35 to get the following from (A3).

$$\forall k \in \{1,\ldots,r\}: \quad \mathbb{E}_{x,l\leftarrow XL}\mathrm{B}^2\left(\psi^{x,l}_{k,B_kY\widetilde{Y}V\widetilde{V}U_l}, \psi^{x,l}_{k,B_kY\widetilde{Y}V\widetilde{V}} \otimes \psi_{U_l}\right) \leq 3\cdot\left(\frac{q}{2^c} + \frac{c\delta}{3} + 2c\delta\right). \tag{5}$$

Intuitively, this claim asserts that for a typical x and ℓ, Bob (conditioned on $X = x$ and $L = \ell$) has very little information about the cell U_ℓ at the end of round k, which is quantified by saying their joint state is close to being a product state. A symmetric statement also follows for Alice. Equation 5 implies the following relation, which is proved in Claim 36: $\Pr_{x,y,l,u_l,v_l\leftarrow X,Y,L,U_L,V_L}[G_l(x,y,u_l,v_l) = \alpha(x,y)] \leq 1/100$, where $\alpha(x,y)$ is either 0 or 1. We then proceed to apply Claim 37.

We then apply Claim 38, which uses (4) and (5) and Claim 37, to obtain the following. There exists, $x, y, l, \widetilde{u}_l, \widetilde{v}_l, \widetilde{\widetilde{u}}_l, \widetilde{\widetilde{v}}_l$ such that,

$$\Delta\left(\psi^{x,y,l,\widetilde{u}_l,\widetilde{v}_l}_{r,A_rU_{-l}\widetilde{U}_{-l}}, \psi^{x,y,l,\widetilde{u}_l,\widetilde{\widetilde{v}}_l}_{r,A_rU_{-l}\widetilde{U}_{-l}}\right) \leq 1000r\cdot\sqrt{\left(\frac{q}{2^c} + \frac{c\delta}{3} + 2c\delta\right)} < 0.1,$$
$$G_l(x,y,\widetilde{u}_l,\widetilde{v}_l) = 1 \text{ and } G_l(x,y,\widetilde{u}_l,\widetilde{\widetilde{v}}_l) = 0. \tag{6}$$

We assume (w.l.o.g) that Alice gives the answer in round r. From above

$$|\Pr(\text{Alice outputs 1 on } (x,y,\widetilde{u}_l,\widetilde{v}_l)) - \Pr(\text{Alice outputs 1 on } (x,y,\widetilde{u}_l,\widetilde{\widetilde{v}}_l))| < 0.1.$$

This is a contradiction since $G_l(x,y,\widetilde{u}_l,\widetilde{v}_l) = 1$ and $G_l(x,y,\widetilde{u}_l,\widetilde{\widetilde{v}}_l) = 0$ and the error of Π on any input is at most 1/3.

Minimax argument. Note that in all branches where we did not reach a contradiction, we constructed a protocol satisfying (3). Hence we constructed, for any μ over $\mathrm{dom}(F)$, a protocol Π_μ that satisfies (3). We now use Fact 21 to complete the proof. ◀

This completes the proof of the theorem, except for the claims Claim 34, Claim 35, Claim 36, Claim 37, and Claim 38 that we did not prove. We now prove these claims.

5.1 Proof of claims

▶ **Claim 34.** *Suppose* $\mathrm{QCC}(\Pi) = 2q$ *and* $\Delta(XL, X \otimes W) \leq \delta_1$. *Then*

$$\mathbb{E}_{x,l \leftarrow XL} \mathrm{B}^2 \left(\psi^x_{k,B_k Y \widetilde{Y} V \widetilde{V} U_l}, \psi^x_{k,B_k Y \widetilde{Y} V \widetilde{V}} \otimes \psi_{U_l} \right) \leq \frac{q}{2^c} + \delta_1.$$

for all $1 \leq k \leq r$. *Here* ψ_{U_l} *is the maximally mixed state on the register* U_l *(in other words a random variable which is uniformly distributed.)*

Proof. We have

$$\begin{aligned}
q &\geq \mathbb{I}(B_k Y \widetilde{Y} V \widetilde{V}; U_0, \ldots, U_{2^c-1} | X)_{\psi_k} && \text{(Lemma 25)} \\
&\geq \sum_{l=0}^{2^c-1} \mathbb{I}(B_k Y \widetilde{Y} V \widetilde{V}; U_l | X)_{\psi_k} && \text{(Fact 17.F)} \\
&= 2^c \cdot \mathbb{E}_{x,l \leftarrow X \otimes W} \mathbb{I}(B_k Y \widetilde{Y} V \widetilde{V}; U_l | X = x)_{\psi_k} \\
&\geq 2^c \cdot \mathbb{E}_{x,l \leftarrow X \otimes W} \mathrm{B}^2 \left(\psi^x_{k,B_k Y \widetilde{Y} V \widetilde{V} U_l}, \psi^x_{k,B_k Y \widetilde{Y} V \widetilde{V}} \otimes \psi_{U_l} \right) && \text{(Fact 17.J).}
\end{aligned}$$

This implies that

$$\mathbb{E}_{x,l \leftarrow X \otimes W} \mathrm{B}^2 \left(\psi^x_{k,B_k Y \widetilde{Y} V \widetilde{V} U_l}, \psi^x_{k,B_k Y \widetilde{Y} V \widetilde{V}} \otimes \psi_{U_l} \right) \leq \frac{q}{2^c}.$$

Since $\Delta(XL, X \otimes W) \leq \delta_1$ and $\mathrm{B}^2(\rho, \sigma) \leq 1$ always, this proves the claim as well. ◀

The next claim intuitively says that, if the communication cost of Π is small, then at any point during the protocol, Bob's register has small information about the correct cheat sheet cell.

▶ **Claim 35.** *Assume in addition to the assumptions of Claim 34, the following condition holds: for all* $i \in [c]$, *let*

$$\mathbb{I}(A_k U \widetilde{U} X_{-i} Y_{-i}; L_i | X_i)_{\psi_k} \leq \delta.$$

Then

$$\mathbb{E}_{x,l \leftarrow XL} \mathrm{B}^2 \left(\psi^{x,l}_{k,B_k Y \widetilde{Y} V \widetilde{V} U_l}, \psi^{x,l}_{k,B_k Y \widetilde{Y} V \widetilde{V}} \otimes \psi_{U_l} \right) \leq 3 \cdot \left(\frac{q}{2^c} + \delta_1 + 2c\delta \right)$$

for all $1 \leq k \leq r$.

Proof. We first prove that the register A_k carries low information about L i.e.

$$\mathbb{I}(A_k U \widetilde{U}; L | X)_{\psi^k} \leq c\delta.$$

This follows from the following chain of inequalities:

$$\begin{aligned}
\delta &\geq \mathbb{I}(A_k U \widetilde{U} X_{-i} Y_{-i}; L_i | X_i)_{\psi_k} \\
&\geq \mathbb{I}(A_k U \widetilde{U} X_{-i} L_{<i}; L_i | X_i)_{\psi_k} && \text{(Fact 17.D and Fact 17.J)} \\
&\geq \mathbb{I}(A_k U \widetilde{U}; L_i | L_{<i}, X)_{\psi_k} && \text{(Fact 17.E).}
\end{aligned}$$

By summing the inequality over i, we get

$$\begin{aligned} c\delta &\geq \sum_{i=1}^{c} \mathbb{I}(A_k U\widetilde{U}; L_i | L_{<i}, X)_{\psi_k} \\ &= \mathbb{I}(A_k U\widetilde{U}; L | X)_{\psi_k} \qquad \text{(Fact 17.C).} \end{aligned}$$

This implies using Fact 17.J:

$$\mathbb{E}_{x,l \leftarrow XL} \mathrm{B}^2 \left(\psi^{x,l}_{k,A_k U\widetilde{U}}, \psi^{x}_{k,A_k U\widetilde{U}} \right) \leq c\delta. \tag{7}$$

Now consider the following two pure states (one conditioned on x, l and the other conditioned on x):

$$\left|\psi^{x,l}\right\rangle_{k,Y\widetilde{Y}V\widetilde{V}U\widetilde{U}A_kB_k} = \sum_{y,v,u} \sqrt{\mu_T(y,v,u|X=x,L=l)} \left|uu\right\rangle_{U\widetilde{U}} \left|yy\right\rangle_{Y\widetilde{Y}} \left|vv\right\rangle_{V\widetilde{V}} \left|\psi^{xuyv}\right\rangle_{k,A_kB_k}$$

and

$$\left|\psi^{x}\right\rangle_{k,Y\widetilde{Y}V\widetilde{V}U\widetilde{U}A_kB_k} = \sum_{y,v,u} \sqrt{\mu_T(y,v,u|X=x)} \left|uu\right\rangle_{U\widetilde{U}} \left|yy\right\rangle_{Y\widetilde{Y}} \left|vv\right\rangle_{V\widetilde{V}} \left|\psi^{x,u,y,v}\right\rangle_{k,A_kB_k}.$$

The marginals of these states on the systems $A_k U\widetilde{U}$ are close as shown above. Now by Uhlmann's theorem (Fact 12), there exists a unitary acting on the systems $B_k Y\widetilde{Y}V\widetilde{V}$ (and the unitary depends on x, l) $\mathcal{U}^{x,l}_{B_kY\widetilde{Y}V\widetilde{V}}$ s.t.

$$\begin{aligned} \mathrm{B}^2 \left(\mathbb{1}_{A_kU\widetilde{U}} \otimes \mathcal{U}^{x,l}_{B_kY\widetilde{Y}V\widetilde{V}} \left|\psi^{x,l}\right\rangle_{k,A_kU\widetilde{U}B_kY\widetilde{Y}V\widetilde{V}}, \left|\psi^{x}\right\rangle_{k,A_kU\widetilde{U}B_kY\widetilde{Y}V\widetilde{V}} \right) \\ = \mathrm{B}^2 \left(\psi^{x,l}_{k,A_kU\widetilde{U}}, \psi^{x}_{k,A_kU\widetilde{U}} \right). \end{aligned} \tag{8}$$

The unitary $\mathcal{U}^{x,l}_{B_kY\widetilde{Y}V\widetilde{V}}$ should be intuitively thought of as implementing the operation of "forgetting L". Hence Equation (7) gives us that:

$$\mathbb{E}_{x,l \leftarrow XL} \mathrm{B}^2 \left(\mathbb{1}_{A_kU\widetilde{U}} \otimes \mathcal{U}^{x,l}_{B_kY\widetilde{Y}V\widetilde{V}} \left|\psi^{x,l}\right\rangle_{k,A_kU\widetilde{U}B_kY\widetilde{Y}V\widetilde{V}}, \left|\psi^{x}\right\rangle_{k,A_kU\widetilde{U}B_kY\widetilde{Y}V\widetilde{V}} \right) \leq c\delta. \tag{9}$$

For all (x, ℓ), define,

$$\phi^{x,\ell} = \mathbb{1}_{A_kU\widetilde{U}} \otimes \mathcal{U}^{x,l}_{B_kY\widetilde{Y}V\widetilde{V}} \left|\psi^{x,l}\right\rangle_{k,A_kU\widetilde{U}B_kY\widetilde{Y}V\widetilde{V}}.$$

Combining Equation (9) with the monotonicity of Bures metric (Fact 14), we obtain the following:

$$\mathbb{E}_{x,l \leftarrow XL} \mathrm{B}^2 \left(\phi^{x,l}_{k,B_kY\widetilde{Y}V\widetilde{V}U_l}, \psi^{x}_{k,B_kY\widetilde{Y}V\widetilde{V}U_l} \right) \leq c\delta \tag{10}$$

and

$$\mathbb{E}_{x,l \leftarrow XL} \mathrm{B}^2 \left(\phi^{x,l}_{k,B_kY\widetilde{Y}V\widetilde{V}}, \psi^{x}_{k,B_kY\widetilde{Y}V\widetilde{V}} \right) \leq c\delta. \tag{11}$$

Furthermore, combining Equation (11) with Fact 13.B, we obtain:

$$\mathbb{E}_{x,l \leftarrow XL} \mathrm{B}^2 \left(\phi^{x,l}_{k,B_kY\widetilde{Y}V\widetilde{V}} \otimes \psi_{U_l}, \psi^{x}_{k,B_kY\widetilde{Y}V\widetilde{V}} \otimes \psi_{U_l} \right) \leq c\delta. \tag{12}$$

Claim 34 gives us that:

$$\mathbb{E}_{x,l\leftarrow XL}\mathrm{B}^2\left(\psi^{x}_{k,B_kY\widetilde{Y}V\widetilde{V}U_l},\psi^{x}_{k,B_kY\widetilde{Y}V\widetilde{V}}\otimes\psi_{U_l}\right)\leq\frac{q}{2^c}+\delta_1. \tag{13}$$

Now combining Equations (10), (12) and (13) along with weak triangle inequality for square of Bures metric (Fact 13.A) and Fact 14, we obtain:

$$\begin{aligned}&\mathbb{E}_{x,l\leftarrow XL}\mathrm{B}^2\left(\psi^{x,l}_{k,B_kY\widetilde{Y}V\widetilde{V}U_l},\psi^{x,l}_{k,B_kY\widetilde{Y}V\widetilde{V}}\otimes\psi_{U_l}\right)\\&=\mathbb{E}_{x,l\leftarrow XL}\mathrm{B}^2\left(\phi^{x,l}_{k,B_kY\widetilde{Y}V\widetilde{V}U_l},\phi^{x,l}_{k,B_kY\widetilde{Y}V\widetilde{V}}\otimes\psi_{U_l}\right)\\&\leq 3\cdot\left(\frac{q}{2^c}+\delta_1+2c\delta\right).\end{aligned}$$

◀

▶ **Claim 36.** *Assuming the conclusion from Claim 35, it holds that*

$$\Pr_{x,y,l,u_l,v_l\leftarrow X,Y,L,U_L,V_L}[G_l(x,y,u_l,v_l)=\alpha(x,y)]\leq 1/100,$$

where $\alpha(x,y)$ is either 0 or 1 for every x,y.

Proof. Using monotonicity and partial measurement (Fact 17.D and Fact 17.B), we have that:

$$\mathbb{E}_{x,y,l,u_l,v_l\leftarrow XYLU_LV_L}\mathrm{B}^2\left(\psi^{x,y,l,u_l,v_l}_{r,B_r},\psi^{x,y,l,v_l}_{r,B_r}\right)\leq 3\cdot\left(\frac{q}{2^c}+\frac{c\delta}{3}+2c\delta\right)$$

Let the output register be called O. Then, from our choice of parameters and monotonicity (Fact 17.D), above inequality implies

$$\mathbb{E}_{x,y,l,u_l,v_l\leftarrow XYLU_LV_L}\mathrm{B}^2\left(\psi^{x,y,l,u_l,v_l}_{r,O},\psi^{x,y,l,v_l}_{r,O}\right)\leq 1/400 \tag{14}$$

Since protocol makes an error of at most 1/400 (which can be assumed due to Fact 20), we have that

$$\mathbb{E}_{x,y,l,u_l,v_l\leftarrow XYLU_LV_L}\mathrm{B}^2(\psi^{x,y,l,u_l,v_l}_{r,O},|G_l(x,y,u_l,v_l)\rangle\langle G_l(x,y,u_l,v_l)|)\leq 1/400. \tag{15}$$

On the other hand, since the look-up function is an XOR family, we find that for a fixed x,y (and hence a fixed l),

$$\begin{aligned}\mathbb{E}_{u_l\leftarrow U_L}|G_l(x,y,u_l,v_l)\rangle\langle G_l(x,y,u_l,v_l)|=&\Pr_{u_l,v_l\leftarrow U_l,V_l|x,y,l}[G_l(x,y,u_l,v_l)=0]\,|0\rangle\langle 0|\\&+\Pr_{u_l,v_l\leftarrow U_l,V_l|x,y,l}[G_l(x,y,u_l,v_l)=1]\,|1\rangle\langle 1|.\end{aligned}$$

Define

$$p^0_{x,y,l}=\Pr_{u_l,v_l\leftarrow U_l,V_l|x,y,l}[G_l(x,y,u_l,v_l)=0],$$

$$p^1_{x,y,l}=\Pr_{u_l,v_l\leftarrow U_l,V_l|x,y,l}[G_l(x,y,u_l,v_l)=1].$$

Then above equation, along with Equation (15) implies that

$$\mathbb{E}_{x,y,l,u_l,v_l\leftarrow XYLU_lV_l}\mathrm{B}^2(\psi^{x,y,l,v_l}_{r,O},p^0_{x,y,l}\,|0\rangle\langle 0|+p^1_{x,y,l}\,|1\rangle\langle 1|)\leq 1/400$$

which in conjunction with Equation 14 and triangle inequality gives us

$$\mathbb{E}_{x,y,l,u_l,v_l\leftarrow XYLU_lV_l}\mathrm{B}^2(|G_l(x,y,u_l,v_l)\rangle\langle G_l(x,y,u_l,v_l)|,p^0_{x,y,l}\,|0\rangle\langle 0|+p^1_{x,y,l}\,|1\rangle\langle 1|)\leq 1/100. \tag{16}$$

This directly implies that we cannot have both $p^0_{x,y,l}, p^1_{x,y,l}$ large. More formally, for every x, y, let $\alpha(x,y)$ be such that $p^{\alpha(x,y)}_{x,y,l} < p^{1-\alpha(x,y)}_{x,y,l}$. Then it is clear that

$$\mathrm{B}^2(|G_l(x,y,u_l,v_l)\rangle\langle G_l(x,y,u_l,v_l)|, p^0_{x,y,l}|0\rangle\langle 0| + p^1_{x,y,l}|1\rangle\langle 1|) > p^{\alpha(x,y)}_{x,y,l},$$

which in turn implies (when used in Equation 16),

$$\mathbb{E}_{x,y,l\leftarrow XYL} p^{\alpha(x,y)}_{x,y,l} = \mathbb{E}_{x,y,l,u_l,v_l\leftarrow XYLU_lV_l} p^{\alpha(x,y)}_{x,y,l} \leq 1/100.$$

Recalling the definition of $p^{\alpha(x,y)}_{x,y,l}$, this immediately gives us

$$\mathbb{E}_{x,y,l\leftarrow XYL}\mathrm{Pr}_{u_l,v_l\leftarrow U_l,V_l|x,y,l}[G_l(x,y,u_l,v_l) = \alpha(x,y)] \leq 1/100.$$

This completes the proof. ◀

▶ **Claim 37.** *Assume that the assumptions of Claim 34 and Claim 35 hold. In addition,*

$$\mathbb{I}(B_kV\widetilde{V}X_{-i}Y_{-i}; L_i|Y_i)_{\psi_k} \leq \delta$$

and

$$\mathrm{Pr}_{x,y,l,u_l,v_l\leftarrow X,Y,L,U_L,V_L}[G_l(x,y,u_l,v_l) = \alpha(x,y)] \leq 1/100$$

also hold for $\alpha(x,y) \in \{0,1\}$ for every x, y. Then there exist $x, y, l = l(x,y), \widetilde{u}_l, \widetilde{v}_l, \widetilde{\widetilde{u}}_l, \widetilde{\widetilde{v}}_l$ s.t. the following conditions hold:

1. $G_l(x,y,\widetilde{u}_l,\widetilde{v}_l) = \alpha(x,y)$.
2. $G_l(x,y,\widetilde{u}_l,\widetilde{\widetilde{v}}_l) = G_l(x,y,\widetilde{\widetilde{u}}_l,\widetilde{\widetilde{v}}_l) = G_l(x,y,\widetilde{\widetilde{u}}_l,\widetilde{v}_l) = 1-\alpha(x,y)$.
3. $\sum_{k=1}^r \mathrm{B}\left(\psi^{x,y,u,v}_{k,B_kV_{-l}\widetilde{V}_{-l}}, \psi^{x,y,v}_{k,B_kV_{-l}\widetilde{V}_{-l}}\right) \leq 80r\cdot\sqrt{\left(\frac{q}{2^c}+\delta_1+2c\delta\right)}$, *for any choice of* $(u,v) = (\widetilde{u}_l,\widetilde{\widetilde{v}}_l), (\widetilde{\widetilde{u}}_l,\widetilde{\widetilde{v}}_l), (\widetilde{\widetilde{u}}_l,\widetilde{v}_l)$.
4. $\sum_{k=1}^r \mathrm{B}\left(\psi^{x,y,u,v}_{k,A_kU_{-l}\widetilde{U}_{-l}}, \psi^{x,y,u}_{k,A_kU_{-l}\widetilde{U}_{-l}}\right) \leq 80r\cdot\sqrt{\left(\frac{q}{2^c}+\delta_1+2c\delta\right)}$, *for any choice of* $(u,v) = (\widetilde{u}_l,\widetilde{\widetilde{v}}_l), (\widetilde{\widetilde{u}}_l,\widetilde{\widetilde{v}}_l), (\widetilde{\widetilde{u}}_l,\widetilde{v}_l)$.

Proof. By Claim 35, we have that for all $1 \leq k \leq r$,

$$\mathbb{E}_{x,l\leftarrow XL}\mathrm{B}^2\left(\psi^{x,l}_{k,B_kY\widetilde{Y}V\widetilde{V}U_l}, \psi^{x,l}_{k,B_kY\widetilde{Y}V\widetilde{V}}\otimes\psi_{U_l}\right) \leq 3\cdot\left(\frac{q}{2^c}+\delta_1+2c\delta\right).$$

By monotonicity of Bures metric (Fact 14), we get that

$$\mathbb{E}_{x,l\leftarrow XL}\mathrm{B}^2\left(\psi^{x,l}_{k,B_kYV_{-l}\widetilde{V}_{-l}U_lV_l}, \psi^{x,l}_{k,B_kYV_{-l}\widetilde{V}_{-l}V_l}\otimes\psi^{U_l}\right) \leq 3\cdot\left(\frac{q}{2^c}+\delta_1+2c\delta\right).$$

Note that in both the states above, the marginal state on registers U_lV_l is maximally mixed. Then by the partial measurement property of the square of Bures metric, Fact 13.C, we get that

$$\mathbb{E}_{x,y,l,u_l,v_l\leftarrow XYLU_LV_L}\mathrm{B}^2\left(\psi^{x,y,l,u_l,v_l}_{k,B_kV_{-l}\widetilde{V}_{-l}}, \psi^{x,y,l,v_l}_{k,B_kV_{-l}\widetilde{V}_{-l}}\right) \leq 3\cdot\left(\frac{q}{2^c}+\delta_1+2c\delta\right).$$

Convexity of square gives us that

$$\mathbb{E}_{x,y,l,u_l,v_l\leftarrow XYLU_LV_L}\mathrm{B}\left(\psi^{x,y,l,u_l,v_l}_{k,B_kV_{-l}\widetilde{V}_{-l}}, \psi^{x,y,l,v_l}_{k,B_kV_{-l}\widetilde{V}_{-l}}\right) \leq \sqrt{3}\cdot\sqrt{\left(\frac{q}{2^c}+\delta_1+2c\delta\right)}. \tag{17}$$

Similarly we get that for all $1 \leq k \leq r$,

$$\mathbb{E}_{x,y,l,u_l,v_l \leftarrow XYLU_LV_L} \mathrm{B}\left(\psi^{x,y,l,u_l,v_l}_{k,A_kU_{-l}\widetilde{U}_{-l}}, \psi^{x,y,l,u_l}_{k,A_kU_{-l}\widetilde{U}_{-l}}\right) \leq \sqrt{3} \cdot \sqrt{\left(\frac{q}{2^c} + \delta_1 + 2c\delta\right)}. \tag{18}$$

Summing Equations (17) and (18) over k, we get the following:

$$\mathbb{E}_{x,y,l,u_l,v_l \leftarrow XYLU_LV_L} \sum_{k=1}^{r} \mathrm{B}\left(\psi^{x,y,l,u_l,v_l}_{k,B_kV_{-l}\widetilde{V}_{-l}}, \psi^{x,y,l,v_l}_{k,B_kV_{-l}\widetilde{V}_{-l}}\right) \leq 2r \cdot \sqrt{\left(\frac{q}{2^c} + \delta_1 + 2c\delta\right)}.$$

and

$$\mathbb{E}_{x,y,l,u_l,v_l \leftarrow XYLU_LV_L} \sum_{k=1}^{r} \mathrm{B}\left(\psi^{x,y,l,u_l,v_l}_{k,A_kU_{-l}\widetilde{U}_{-l}}, \psi^{x,y,l,u_l}_{k,A_kU_{-l}\widetilde{U}_{-l}}\right) \leq 2r \cdot \sqrt{\left(\frac{q}{2^c} + \delta_1 + 2c\delta\right)}.$$

Now by Markov's inequality, we can find $x, y, l = l(x, y)$ s.t. the following hold:

$$\Pr_{u_l,v_l \leftarrow U_l,V_l}[G_l(x, y, u_l, v_l) = \alpha(x, y)] \leq 1/25, \tag{19}$$

$$\mathbb{E}_{u_l,v_l \leftarrow U_lV_l} \sum_{k=1}^{r} \mathrm{B}\left(\psi^{x,y,l,u_l,v_l}_{k,B_kV_{-l}\widetilde{V}_{-l}}, \psi^{x,y,l,v_l}_{k,B_kV_{-l}\widetilde{V}_{-l}}\right) \leq 8r \cdot \sqrt{\left(\frac{q}{2^c} + \delta_1 + 2c\delta\right)}, \tag{20}$$

$$\mathbb{E}_{u_l,v_l \leftarrow U_lV_l} \sum_{k=1}^{r} \mathrm{B}\left(\psi^{x,y,l,u_l,v_l}_{k,A_kU_{-l}\widetilde{U}_{-l}}, \psi^{x,y,l,u_l}_{k,A_kU_{-l}\widetilde{U}_{-l}}\right) \leq 8r \cdot \sqrt{\left(\frac{q}{2^c} + \delta_1 + 2c\delta\right)}. \tag{21}$$

Without loss of generality, assume that $\alpha(x, y) = 1$. Let us have the following two notations:

$$\kappa_A(u_l, v_l) := \sum_{k=1}^{r} \mathrm{B}\left(\psi^{x,y,l,u_l,v_l}_{k,A_kU_{-l}\widetilde{U}_{-l}}, \psi^{x,y,l,u_l}_{k,A_kU_{-l}\widetilde{U}_{-l}}\right),$$

$$\kappa_{\mathrm{B}}(u_l, v_l) := \sum_{k=1}^{r} \mathrm{B}\left(\psi^{x,y,l,u_l,v_l}_{k,B_kV_{-l}\widetilde{V}_{-l}}, \psi^{x,y,l,v_l}_{k,B_kV_{-l}\widetilde{V}_{-l}}\right).$$

Recall that for $l = l(x, y)$, $G_l(x, y, u_l, v_l)$ is a non-trivial XOR function of the inputs u_l, v_l. So there exists a $t \in \{0,1\}^m$ s.t. $G_l(x, y, u, u \oplus t) = 1$ for all $u \in \{0,1\}^m$. Now we will choose $\widetilde{u}_l, \widetilde{\widetilde{u}}_l, \widetilde{\widetilde{v}}_l$ uniformly and independently from $\{0,1\}^m$ and set $\widetilde{v}_l = \widetilde{u}_l \oplus t$. Note that marginally, the distribution of (u, v) is uniform over $\{0,1\}^m \times \{0,1\}^m$, for any choice of $(u, v) = (\widetilde{u}_l, \widetilde{\widetilde{v}}_l), (\widetilde{\widetilde{u}}_l, \widetilde{\widetilde{v}}_l), (\widetilde{\widetilde{u}}_l, \widetilde{v}_l)$. Hence for any choice of $(u, v) = (\widetilde{u}_l, \widetilde{\widetilde{v}}_l), (\widetilde{\widetilde{u}}_l, \widetilde{\widetilde{v}}_l), (\widetilde{\widetilde{u}}_l, \widetilde{v}_l)$, from Equations (19), (20) and (21), we get the following:

$$\Pr_{\substack{\sim \\ u_l,u_l,v_l}} \approx \approx [G_l(x, y, u, v) = 1] \leq 1/25,$$

$$\mathbb{E}_{\substack{\sim \\ u_l,u_l,v_l}} \approx \approx \kappa_A(u, v) \leq 8r \cdot \sqrt{\left(\frac{q}{2^c} + \delta_1 + 2c\delta\right)},$$

$$\mathbb{E}_{\substack{\sim \\ u_l,u_l,v_l}} \approx \approx \kappa_{\mathrm{B}}(u, v) \leq 8r \cdot \sqrt{\left(\frac{q}{2^c} + \delta_1 + 2c\delta\right)}.$$

Now by a simple application of Markov's inequality, there exists a setting of $(\widetilde{u}_l, \widetilde{\widetilde{u}}_l, \widetilde{\widetilde{v}}_l)$ so that for any choice of $(u, v) = (\widetilde{u}_l, \widetilde{\widetilde{v}}_l), (\widetilde{\widetilde{u}}_l, \widetilde{\widetilde{v}}_l), (\widetilde{\widetilde{u}}_l, \widetilde{v}_l)$,

$$G_l(x, y, u, v) = 0,$$

$$\kappa_A(u, v) \leq 80r \cdot \sqrt{\left(\frac{q}{2^c} + \delta_1 + 2c\delta\right)},$$

$$\kappa_{\mathrm{B}}(u, v) \leq 80r \cdot \sqrt{\left(\frac{q}{2^c} + \delta_1 + 2c\delta\right)}.$$

This completes the proof. Note that we chose $\widetilde{v}_l$ so that $G_l(x, y, \widetilde{u}_l, \widetilde{v}_l) = 1$. ◀

The next claim will follow from the quantum-cut-and-paste lemma applied to Claim 35.

▶ **Claim 38.** *Assume that the assumptions of Claim 34, Claim 35 and Claim 37 hold. Then for the $x, y, l, \widetilde{u}_l, \widetilde{v}_l, \widetilde{\widetilde{u}}_l, \widetilde{\widetilde{v}}_l$ in Claim 37, it holds that*

$$\Delta\left((\psi^{x,y,l,\widetilde{u}_l,\widetilde{v}_l}_{r,A_rU_{-l}\widetilde{U}_{-l}}, \psi^{x,y,l,\widetilde{u}_l,\widetilde{\widetilde{v}}_l}_{r,A_rU_{-l}\widetilde{U}_{-l}}\right) \leq 1000r \cdot \sqrt{\left(\frac{q}{2^c} + \delta_1 + 2c\delta\right)}.$$

Proof. Let us define the following registers: $\widetilde{A}_k := A_kU_{-l}\widetilde{U}_{-l}$ and $\widetilde{B}_k := B_kV_{-l}\widetilde{V}_{-l}$. Also we will define the following:

$$\delta_{k,A} := \mathrm{B}\left(\psi^{x,y,\widetilde{\widetilde{u}}_l,\widetilde{v}_l}_{k,\widetilde{A}_k}, \psi^{x,y,\widetilde{\widetilde{u}}_l,\widetilde{\widetilde{v}}_l}_{k,\widetilde{A}_k}\right),$$
$$\delta_{k,B} := \mathrm{B}\left(\psi^{x,y,\widetilde{u}_l,\widetilde{\widetilde{v}}_l}_{k,\widetilde{B}_k}, \psi^{x,y,\widetilde{\widetilde{u}}_l,\widetilde{\widetilde{v}}_l}_{k,\widetilde{B}_k}\right).$$

By the triangle inequality for Bures metric Fact 13.A,

$$\delta_{k,A} \leq \mathrm{B}\left(\psi^{x,y,\widetilde{\widetilde{u}}_l,\widetilde{v}_l}_{k,\widetilde{A}_k}, \psi^{x,y,\widetilde{\widetilde{u}}_l}_{k,\widetilde{A}_k}\right) + \mathrm{B}\left(\psi^{x,y,\widetilde{\widetilde{u}}_l,\widetilde{\widetilde{v}}_l}_{k,\widetilde{A}_k}, \psi^{x,y,\widetilde{\widetilde{u}}_l}_{k,\widetilde{A}_k}\right), \tag{22}$$
$$\delta_{k,B} \leq \mathrm{B}\left(\psi^{x,y,\widetilde{u}_l,\widetilde{\widetilde{v}}_l}_{k,\widetilde{B}_k}, \psi^{x,y,\widetilde{\widetilde{v}}_l}_{k,\widetilde{B}_k}\right) + \mathrm{B}\left(\psi^{x,y,\widetilde{\widetilde{u}}_l,\widetilde{\widetilde{v}}_l}_{k,\widetilde{A}_k}, \psi^{x,y,\widetilde{\widetilde{v}}_l}_{k,\widetilde{A}_k}\right). \tag{23}$$

Combining Equations (22), (23) and Claim 37, we get the following:

$$\sum_{k=1}^{r} \delta_{k,A} \leq 160r \cdot \sqrt{\left(\frac{q}{2^c} + \delta_1 + 2c\delta\right)},$$
$$\sum_{k=1}^{r} \delta_{k,B} \leq 160r \cdot \sqrt{\left(\frac{q}{2^c} + \delta_1 + 2c\delta\right)}.$$

Note that the state $\psi^{x,y,u,v}_{k,\widetilde{A}_k,\widetilde{B}_k}$ is a pure state for every k, x, y, u, v. Also for a fixed x, y, these states can be formed by a quantum protocol Π' where Alice gets the input u and Bob gets the input v (since they are originally formed by running the protocol Π and $U_{-l}\widetilde{U}_{-l}$ and $V_{-l}\widetilde{V}_{-l}$ are registers that can be owned by Alice and Bob respectively at the start of Π'). Hence we can apply Lemma 24 (by setting $u = \widetilde{\widetilde{u}}_l$, $u' = \widetilde{u}_l$, $v = \widetilde{\widetilde{v}}_l$, $v' = \widetilde{v}_l$) to conclude that

$$\begin{aligned}\mathrm{B}\left(\psi^{x,y,l,\widetilde{u}_l,\widetilde{v}_l}_{r,\widetilde{A}_r}, \psi^{x,y,l,\widetilde{u}_l,\widetilde{\widetilde{v}}_l}_{r,\widetilde{A}_r}\right) &\leq 2\sum_{k=1}^{r}(\delta_{k,A} + \delta_{k,B}) \\ &\leq 640r \cdot \sqrt{\left(\frac{q}{2^c} + \delta_1 + 2c\delta\right)}.\end{aligned}$$

Now the proof is finished by Fact 11 and monotonicity of trace distance (Fact 14). ◀

6 Conclusion and open problems

We prove a nearly quadratic separation between the log of approximate rank and quantum communication complexity for a family of total functions, which is also the first superlinear separation between these two measures. Our separation is based on a lookup function constructed from the inner product function. To prove the lower bound on the quantum communication complexity of this lookup function, we prove a general purpose cheat sheet theorem for quantum communication complexity. We also prove a general theorem about

an upper bound on log of approximate rank of lookup functions based on the circuit size of the base function. This proves the upper bound for an appropriate lookup function on inner product because the inner product function has a linear size circuit.

Several interesting open problems arise out of our work. We state some of them here:

1. Can we eliminate the round dependence in Theorem 33? Can we prove a similar result for quantum information complexity instead of quantum communication complexity, thereby separating quantum information complexity from log of approximate rank?
2. Can we separate the quantum partition bound [29] from quantum communication complexity? Is the quantum partition bound a stronger lower bound measure than log of approximate rank?
3. Can we prove some sort of cheat sheet theorem for log of approximate rank? A simpler question might be to prove that for the inner product function on n bits, any lookup function contructed using a nontrivial XOR family of functions has log of approximate rank at least $\Omega(\sqrt{n})$.

Acknowledgments. We thank Aleksandrs Belovs, Mika Göös, and Miklos Santha for interesting discussions during the writing of [5].

References

1 Scott Aaronson, Shalev Ben-David, and Robin Kothari. Separations in query complexity using cheat sheets. In *Proceedings of the 48th ACM Symposium on Theory of Computing (STOC 2016)*, pages 863–876, 2016. doi:10.1145/2897518.2897644.

2 Andris Ambainis. Polynomial degree vs. quantum query complexity. In *Proceedings of the 44th IEEE Symposium on Foundations of Computer Science (FOCS 2003)*, pages 230–239, 2003. doi:10.1109/SFCS.2003.1238197.

3 Andris Ambainis, Kaspars Balodis, Aleksandrs Belovs, Troy Lee, Miklos Santha, and Juris Smotrovs. Separations in query complexity based on pointer functions. In *Proceedings of the 48th ACM Symposium on Theory of Computing (STOC 2016)*, 2016. doi:10.1145/2897518.2897524.

4 Andris Ambainis, Martins Kokainis, and Robin Kothari. Nearly Optimal Separations Between Communication (or Query) Complexity and Partitions. In *31st Conference on Computational Complexity (CCC 2016)*, volume 50 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 4:1–4:14, Dagstuhl, Germany, 2016. Schloss Dagstuhl – Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.CCC.2016.4.

5 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. *Proceedings of the 57h IEEE Symposium on Foundations of Computer Science (FOCS 2016)*, 2016. arXiv preprint arXiv:1605.01142.

6 Koenraad M. R. Audenaert. Quantum skew divergence. *Journal of Mathematical Physics*, 55(11), 2014. doi:10.1063/1.4901039.

7 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.

8 Howard Barnum, Carlton M. Caves, Christopher A. Fuchs, Richard Jozsa, and Benjamin Schmacher. Noncommuting mixed states cannot be broadcast. *Phys. Rev. Lett.*, 76(15):2818–2821, 1996. doi:10.1103/PhysRevLett.76.2818.

9 M. Braverman, A. Rao, O. Weinstein, and A. Yehudayoff. Direct products in communication complexity. In *54th Annual Symposium on Foundations of Computer Science (FOCS 2013)*, pages 746–755, Oct 2013. doi:10.1109/FOCS.2013.85.

10 Mark Braverman, Ankit Garg, Young Kun Ko, Jieming Mao, and Dave Touchette. Near-optimal bounds on bounded-round quantum communication complexity of disjointness. In

Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on, pages 773–791, Oct 2015. doi:10.1109/FOCS.2015.53.

11 Mark Braverman and Omri Weinstein. An interactive information odometer and applications. In *Proceedings of the 47th Annual ACM on Symposium on Theory of Computing*, STOC'15, pages 341–350, 2015. doi:10.1145/2746539.2746548.

12 Harry Buhrman, Richard Cleve, and Avi Wigderson. Quantum vs. classical communication and computation. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, STOC'98, pages 63–68, 1998. doi:10.1145/276698.276713.

13 Harry Buhrman and Ronald de Wolf. Communication complexity lower bounds by polynomials. In *Proceedings of the 16th IEEE Conference on Computational Complexity*, pages 120–130. IEEE, 2001. doi:10.1109/CCC.2001.933879.

14 Donald Bures. An extension of Kakutani's theorem on infinite product measures to the tensor product of semifinite ω^*-algebras. *Transactions of the American Mathematical Society*, 135:199–212, 1969. doi:10.2307/1995012.

15 Richard Cleve, Wim van Dam, Michael Nielsen, and Alain Tapp. Quantum entanglement and the communication complexity of the inner product function. *Theoretical Computer Science*, 486:11–19, 2013. doi:10.1016/j.tcs.2012.12.012.

16 Andrew Drucker. *The Complexity of Joint Computation.* PhD thesis, Massachusetts Institute of Technology, 2012.

17 Jürgen Forster. A linear lower bound on the unbounded error probabilistic communication complexity. *Journal of Computer and System Sciences*, 65(4):612–625, 2002. Special Issue on Complexity 2001. doi:10.1016/S0022-0000(02)00019-3.

18 Christopher A. Fuchs and Jeroen van de Graaf. Cryptographic distinguishability measures for quantum-mechanical states. *IEEE Transactions on Information Theory*, 45(4):1216–1227, May 1999. doi:10.1109/18.761271.

19 Mika Göös, T. S. Jayram, Toniann Pitassi, and Thomas Watson. Randomized communication vs. partition number. *Electronic Colloquium on Computational Complexity (ECCC) TR15-169*, 2015.

20 Mika Göös, Toniann Pitassi, and Thomas Watson. Deterministic communication vs. partition number. In *Proceedings of the 56th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 1077–1088, 2015. doi:10.1109/FOCS.2015.70.

21 Lov K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the 28th ACM Symposium on Theory of Computing (STOC)*, pages 212–219, 1996. doi:10.1145/237814.237866.

22 Rahul Jain and Ashwin Nayak. Accessible versus Holevo information for a binary random variable. Preprint, 2006. arXiv:quant-ph/0603278.

23 Rahul Jain, Jaikumar Radhakrishnan, and Pranab Sen. A lower bound for the bounded round quantum communication complexity of set disjointness. In *Proceedings of the 44th IEEE Symposium on Foundations of Computer Science (FOCS 2003)*, pages 220–229, Oct 2003. doi:10.1109/SFCS.2003.1238196.

24 Hartmut Klauck. Lower bounds for quantum communication complexity. *SIAM Journal on Computing*, 37(1):20–46, 2007. doi:10.1137/S0097539702405620.

25 Gillat Kol, Shay Moran, Amir Shpilka, and Amir Yehudayoff. Approximate nonnegative rank is equivalent to the smooth rectangle bound. In *Automata, Languages, and Programming: 41st International Colloquium (ICALP 2014)*, pages 701–712. Springer Berlin Heidelberg, 2014. doi:10.1007/978-3-662-43948-7_58.

26 Ilan Kremer. Quantum communication. Master's thesis, The Hebrew University of Jerusalem, 1995. URL: http://www.cs.huji.ac.il/~noam/kremer-thesis.ps.

27 Eyal Kushilevitz and Noam Nisan. *Communication Complexity.* Cambridge University Press, 2006.

28 Sophie Laplante, Troy Lee, and Mario Szegedy. The quantum adversary method and classical formula size lower bounds. *Computational Complexity*, 15:163–196, 2006. doi:10.1007/s00037-006-0212-7.

29 Sophie Laplante, Virginie Lerays, and Jérémie Roland. Classical and quantum partition bound and detector inefficiency. In *Proceedings of the 39th International Colloquium Conference on Automata, Languages, and Programming – Volume Part I*, ICALP'12, pages 617–628, Berlin, Heidelberg, 2012. Springer-Verlag. doi:10.1007/978-3-642-31594-7_52.

30 Troy Lee and Jérémie Roland. A strong direct product theorem for quantum query complexity. *Computational Complexity*, 22(2):429–462, 2013. doi:10.1007/s00037-013-0066-8.

31 Troy Lee and Adi Shraibman. An approximation algorithm for approximation rank. In *Proceedings of the 24th IEEE Conference on Computational Complexity*, pages 351–357, 2008. doi:10.1109/CCC.2009.25.

32 Göran Lindblad. Completely positive maps and entropy inequalities. *Communications in Mathematical Physics*, 40(2):147–151, 1975. doi:10.1007/BF01609396.

33 Nati Linial and Adi Shraibman. Lower bounds in communication complexity based on factorization norms. *Random Structures & Algorithms*, 34(3):368–394, 2009. doi:10.1002/rsa.20232.

34 Ashwin Nayak and Dave Touchette. Augmented index and quantum streaming algorithms for DYCK(2). *arXiv preprint* arXiv:1610.04937, 2016.

35 Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge Series on Information and the Natural Sciences. CUP, 2000.

36 Alexander Razborov. Quantum communication complexity of symmetric predicates. *Izvestiya: Mathematics*, 67(1):145, 2003. doi:10.1070/IM2003v067n01ABEH000422.

37 Ben W. Reichardt. Reflections for quantum query algorithms. In *Proceedings of the 22nd ACM-SIAM Symposium on Discrete Algorithms (SODA 2011)*, SODA'11, pages 560–569, 2011. URL: http://dl.acm.org/citation.cfm?id=2133036.2133080.

38 Alexander A. Sherstov. The pattern matrix method. *SIAM Journal on Computing*, 40(6):1969–2000, 2011. doi:10.1137/080733644.

39 Alexander A. Sherstov. Strong direct product theorems for quantum communication and query complexity. *SIAM Journal on Computing*, 41(5):1122–1165, 2012. doi:10.1137/110842661.

40 Yaoyun Shi and Yufan Zhu. Quantum communication complexity of block-composed functions. *Quantum information and computation*, 9(5,6):444–460, 2009. arXiv:0710.0095.

41 Maurice Sion. On general minimax theorems. *Pacific J. of Mathematics*, 1:171–176, 1958.

42 Marco Tomamichel. *Quantum Information Processing with Finite Resources: Mathematical Foundations*. SpringerBriefs in Mathematical Physics. Springer, 2016. doi:10.1007/978-3-319-21891-5.

43 Dave Touchette. Quantum information complexity. In *Proceedings of the 47th Annual ACM on Symposium on Theory of Computing*, STOC'15, pages 317–326. ACM, 2015. doi:10.1145/2746539.2746613.

44 A. Uhlmann. The "transition probability" in the state space of a *-algebra. *Reports on Mathematical Physics*, 9:273–279, 1976. doi:10.1016/0034-4877(76)90060-4.

45 John Watrous. *Theory of Quantum Information*. Unpublished, January 2016. Available at https://cs.uwaterloo.ca/~watrous/TQI/.

46 Mark M. Wilde. *Quantum Information Theory*. Cambridge University Press, Cambridge, 12 2012. doi:10.1017/CBO9781139525343.

47 Andrew Yao. Quantum circuit complexity. In *Proceedings of the 34th IEEE Symposium on Foundations of Computer Science (FOCS 1993)*, pages 352–360, 1993. doi:10.1109/SFCS.1993.366852.

Optimal Quantum Sample Complexity of Learning Algorithms*

Srinivasan Arunachalam[1] and Ronald de Wolf[2]

1 **QuSoft – Research Center for Quantum Software, Amsterdam, The Netherlands; and**
CWI – Centrum Wiskunde & Informatica, Amsterdam, The Netherlands; and
University of Amsterdam, Amsterdam, The Netherlands
arunacha@cwi.nl

2 **QuSoft – Research Center for Quantum Software, Amsterdam, The Netherlands; and**
CWI – Centrum Wiskunde & Informatica, Amsterdam, The Netherlands; and
University of Amsterdam, Amsterdam, The Netherlands
rdewolf@cwi.nl

Abstract

In learning theory, the *VC dimension* of a concept class $\mathcal{C}$ is the most common way to measure its "richness." A fundamental result says that the number of examples needed to learn an unknown target concept $c \in \mathcal{C}$ under an unknown distribution D, is tightly determined by the VC dimension d of the concept class $\mathcal{C}$. Specifically, in the PAC model

$$\Theta\Big(\frac{d}{\varepsilon} + \frac{\log(1/\delta)}{\varepsilon}\Big)$$

examples are necessary and sufficient for a learner to output, with probability $1-\delta$, a hypothesis h that is ε-close to the target concept c (measured under D). In the related *agnostic* model, where the samples need not come from a $c \in \mathcal{C}$, we know that

$$\Theta\Big(\frac{d}{\varepsilon^2} + \frac{\log(1/\delta)}{\varepsilon^2}\Big)$$

examples are necessary and sufficient to output an hypothesis $h \in \mathcal{C}$ whose error is at most ε worse than the error of the best concept in $\mathcal{C}$.

Here we analyze *quantum* sample complexity, where each example is a coherent quantum state. This model was introduced by Bshouty and Jackson [18], who showed that quantum examples are more powerful than classical examples in some fixed-distribution settings. However, Atıcı and Servedio [10], improved by Zhang [55], showed that in the PAC setting (where the learner has to succeed for every distribution), quantum examples cannot be much more powerful: the required number of quantum examples is

$$\Omega\Big(\frac{d^{1-\eta}}{\varepsilon} + d + \frac{\log(1/\delta)}{\varepsilon}\Big) \text{ for arbitrarily small constant } \eta > 0.$$

Our main result is that quantum and classical sample complexity are in fact equal up to constant factors in both the PAC and agnostic models. We give two proof approaches. The first is a fairly simple information-theoretic argument that yields the above two classical bounds and yields the same bounds for quantum sample complexity up to a $\log(d/\varepsilon)$ factor. We then give a second approach that avoids the log-factor loss, based on analyzing the behavior of the "Pretty Good Measurement" on the quantum state identification problems that correspond to learning. This

* S.A. was supported by ERC Consolidator Grant QPROGRESS. R.dW. was partially supported by ERC Consolidator Grant QPROGRESS.

32nd Computational Complexity Conference (CCC 2017).
Editor: Ryan O'Donnell; Article No. 25; pp. 25:1–25:31

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

shows classical and quantum sample complexity are equal up to constant factors for every concept class $\mathcal{C}$.

1998 ACM Subject Classification F.1.1 Models of Computation, I.2.6 Learning

Keywords and phrases Quantum computing, PAC learning, agnostic learning, VC dimension

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.25

1 Introduction

1.1 Sample complexity and VC dimension

Machine learning is one of the most successful parts of AI, with impressive practical applications in areas ranging from image processing, speech recognition, to even beating Go champions. Its theoretical aspects have been deeply studied, revealing beautiful structure and mathematical characterizations of when (efficient) learning is or is not possible in various settings.

1.1.1 The PAC setting

Leslie Valiant's Probably Approximately Correct (PAC) model [49] gives a precise complexity-theoretic definition of what it means for a concept class to be (efficiently) learnable. For simplicity we will (without loss of generality) focus on concepts that are Boolean functions, $c : \{0,1\}^n \to \{0,1\}$. Equivalently, a concept c is a subset of $\{0,1\}^n$, namely $\{x : c(x) = 1\}$. Let $\mathcal{C} \subseteq \{f : \{0,1\}^n \to \{0,1\}\}$ be a concept class. This could for example be the class of functions computed by disjunctive normal form (DNF) formulas of a certain size, or Boolean circuits or decision trees of a certain depth.

The goal of a learning algorithm (the learner) is to probably approximate some unknown *target concept* $c \in \mathcal{C}$ from random *labeled examples*. Each labeled example is of the form $(x, c(x))$ where x is distributed according to some unknown distribution D over $\{0,1\}^n$. After processing a number of such examples (hopefully not too many), the learner outputs some *hypothesis* h. We say that h is *ε-approximately correct* (w.r.t. the target concept c) if its error probability under D is at most ε: $\Pr_{x\sim D}[h(x) \neq c(x)] \leq \varepsilon$. Note that the learning phase and the evaluation phase (i.e., whether a hypothesis is approximately correct) are according to the same distribution D – as if the learner is taught and then tested by the same teacher. An (ε, δ)-learner for the concept class $\mathcal{C}$ is one whose hypothesis is probably approximately correct:

> For all target concepts $c \in \mathcal{C}$ and distributions D:
> Pr[the learner's output h is ε-approximately correct] $\geq 1 - \delta$,

where the probability is over the sequence of examples and the learner's internal randomness. Note that we leave the learner the freedom to output an h which is not in $\mathcal{C}$. If always $h \in \mathcal{C}$, then the learner is called a *proper* PAC-learner.

Of course, we want the learner to be as efficient as possible. Its *sample complexity* is the worst-case number of examples it uses, and its *time complexity* is the worst-case running time of the learner. In this paper we focus on sample complexity. This allows us to ignore technical issues of how the runtime of an algorithm is measured, and in what form the hypothesis h is given as output by the learner.

The sample complexity of a concept class $\mathcal{C}$ is the sample complexity of the most efficient learner for $\mathcal{C}$. It is a function of ε, δ, and of course of $\mathcal{C}$ itself. One of the most fundamental results in learning theory is that the sample complexity of $\mathcal{C}$ is tightly determined by a combinatorial parameter called the *VC dimension* of $\mathcal{C}$, due to and named after Vapnik and Chervonenkis [50]. The VC dimension of $\mathcal{C}$ is the size of the biggest $\mathcal{S} \subseteq \{0,1\}^n$ that can be labeled in all $2^{|\mathcal{S}|}$ possible ways by concepts from $\mathcal{C}$: for each sequence of $|\mathcal{S}|$ binary labels for the elements of $\mathcal{S}$, there is a $c \in \mathcal{C}$ that has that labeling (such an $\mathcal{S}$ is said to be *shattered* by $\mathcal{C}$). Knowing this VC dimension (and ε, δ) already tells us the sample complexity of $\mathcal{C}$ up to constant factors. Blumer et al. [17] proved that the sample complexity of $\mathcal{C}$ is lower bounded by $\Omega(d/\varepsilon + \log(1/\delta)/\varepsilon)$, and they proved an upper bound that was worse by a $\log(1/\varepsilon)$-factor. In very recent work, Hanneke [27] (improving on Simon [47]) got rid of this $\log(1/\varepsilon)$-factor for PAC learning,[1] showing that the lower bound of Blumer et al. is in fact optimal: the sample complexity of $\mathcal{C}$ in the PAC setting is

$$\Theta\Big(\frac{d}{\varepsilon} + \frac{\log(1/\delta)}{\varepsilon}\Big). \tag{1}$$

1.1.2 The agnostic setting

The PAC model assumes that the labeled examples are generated according to a target concept $c \in \mathcal{C}$. However, in many learning situations that is not a realistic assumption, for example when the examples are noisy in some way or when we have no reason to believe there is an underlying target concept at all. The *agnostic* model of learning, introduced by Haussler [31] and Kearns et al. [36], takes this into account. Here, the examples are generated according to a distribution D on $\{0,1\}^{n+1}$. The error of a specific concept $c : \{0,1\}^n \to \{0,1\}$ is defined to be $\mathrm{err}_D(c) = \Pr_{(x,b)\sim D}[c(x) \neq b]$. When we are restricted to hypotheses in $\mathcal{C}$, we would like to find the hypothesis that minimizes $\mathrm{err}_D(c)$ over all $c \in \mathcal{C}$. However, it may require very many examples to do that exactly. In the spirit of the PAC model, the goal of the learner is now to output an $h \in \mathcal{C}$ whose error is at most an additive ε worse than that of the best (= lowest-error) concepts in $\mathcal{C}$.

Like in the PAC model, the optimal sample complexity of such agnostic learners is tightly determined by the VC dimension of $\mathcal{C}$: it is

$$\Theta\Big(\frac{d}{\varepsilon^2} + \frac{\log(1/\delta)}{\varepsilon^2}\Big), \tag{2}$$

where the lower bound was proven by Vapnik and Chervonenkis [51] (see also Simon [46]), and the upper bound was proven by Talagrand [48]. Shalev-Shwartz and Ben-David [45, Section 6.4] call Eq. (1) and Eq. (2) the "Fundamental Theorem of PAC learning."

1.2 Our results

In this paper we are interested in *quantum* sample complexity. Here a *quantum example* for some concept $c : \{0,1\}^n \to \{0,1\}$, according to some distribution D, corresponds to an $(n+1)$-qubit state

$$\sum_{x\in\{0,1\}^n} \sqrt{D(x)}|x, c(x)\rangle.$$

[1] Hanneke's learner is not proper, meaning that its hypothesis h is not always in $\mathcal{C}$. It is still an open question whether the $\log(1/\varepsilon)$-factor can be removed for proper PAC learning. Our lower bounds in this paper hold for all learners, quantum as well as classical, and proper as well as improper.

In other words, instead of a random labeled example, an example is now given by a coherent quantum superposition where the square-roots of the probabilities become the amplitudes.[2] This model was introduced by Bshouty and Jackson [18], who showed that DNF formulas are learnable in polynomial time from quantum examples when D is uniform. For learning DNF under the uniform distribution from *classical* examples, the best upper bound is quasipolynomial time [52]. With the added power of "membership queries," where the learner can actively ask for the label of any x of his choice, DNF formulas are known to be learnable in polynomial time under uniform D [33], but *without* membership queries polynomial-time learnability is a longstanding open problem (see [20] for a recent hardness result).

How reasonable are examples that are given as a coherent superposition rather than as a random sample? They may seem unreasonable a priori because quantum superpositions seem very fragile and are easily collapsed by measurement, but if we accept the "church of the larger Hilbert space" view on quantum mechanics, where the universe just evolves unitarily without any collapses, then they may become more palatable. It is also possible that the quantum examples are generated by some coherent quantum process that acts like the teacher.

How many quantum examples are needed to learn a concept class $\mathcal{C}$ of VC dimension d? Since a learner can just measure a quantum example in order to obtain a classical example, the *upper* bounds on classical sample complexity trivially imply the same upper bounds on quantum sample complexity. But what about the lower bounds? Are there situations where quantum examples are more powerful than classical? Indeed there are. We already mentioned the results of Bshouty and Jackson [18] for learning DNF under the uniform distribution without membership queries. Another good example is the learnability of the concept class of linear functions over $\mathbb{F}_2$, $\mathcal{C} = \{c(x) = a \cdot x : a \in \{0,1\}^n\}$, again under the uniform distribution D. It is easy to see that a classical learner needs about n examples to learn an unknown $c \in \mathcal{C}$ under this D. However, if we are given one quantum example

$$\sum_{x\in\{0,1\}^n} \sqrt{D(x)}|x, c(x)\rangle = \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x, a \cdot x\rangle,$$

then a small modification of the Bernstein-Vazirani algorithm [16] can recover a (and hence c) with probability $1/2$. Hence $O(1)$ quantum examples suffice to learn c exactly, with high probability, under the uniform distribution. Atıcı and Servedio [11] used similar ideas to learning *k-juntas* (concepts depending on only k of their n variables) from quantum examples under the uniform distribution. However, PAC learning requires a learner to learn c under *all possible* distributions D, not just the uniform one. The success probability of the Bernstein-Vazirani algorithm deteriorates sharply when D is far from uniform, but that does not rule out the existence of other quantum learners that use $o(n)$ quantum examples and succeed for all D.

Our main result in this paper is that quantum examples are not actually more powerful than classical labeled examples in the PAC model and in the agnostic model: we prove that the lower bounds on classical sample complexity of Eq. (1) and Eq. (2) hold for quantum examples as well. Accordingly, despite several distribution-specific speedups, quantum examples do not significantly reduce sample complexity if we require our learner to work for

[2] We could allow more general quantum examples $\sum_{x\in\{0,1\}^n} \alpha_x|x, c(x)\rangle$, where we only require $|\alpha_x|^2 = D(x)$. However, that will not affect our results since our lower bounds apply to quantum examples where we know the amplitudes are square-rooted probabilities. Adding more degrees of freedom to quantum examples does not make learning easier.

all distributions D. This should be contrasted with the situation when considering the *time complexity* of learning. Servedio and Gortler [44] considered a concept class (already known in the literature [37, Chapter 6]) that can be PAC-learned in polynomial time by a quantum computer, even with only classical examples, but that cannot be PAC-learned in polynomial time by a classical learner unless Blum integers can be factored in polynomial time (which is widely believed to be false).

Earlier work on quantum sample complexity had already gotten close to extending the lower bound of Eq. (1) to PAC learning from quantum examples. Atıcı and Servedio [10] first proved a lower bound of $\Omega(\sqrt{d}/\varepsilon + d + \log(1/\delta)/\varepsilon)$ using the so-called "hybrid method." Their proof technique was subsequently pushed further by Zhang [55] to

$$\Omega\Big(\frac{d^{1-\eta}}{\varepsilon} + d + \frac{\log(1/\delta)}{\varepsilon}\Big) \text{ for arbitrarily small constant } \eta > 0. \tag{3}$$

Here we optimize these bounds, removing the η and achieving the optimal lower bound for quantum sample complexity in the PAC model (Eq. (1)).

We also show that the lower bound (Eq. (2)) for the agnostic model extends to quantum examples. As far as we know, in contrast to the PAC model, no earlier results were known for quantum sample complexity in the agnostic model.

We have two different proof approaches, which we sketch below.

1.2.1 An information-theoretic argument

In Section 3 we give a fairly intuitive information-theoretic argument that gives optimal lower bounds for classical sample complexity, and that gives nearly-optimal lower bounds for quantum sample complexity. Let us first see how we can prove the classical PAC lower bound of Eq. (1). Suppose $\mathcal{S} = \{s_0, s_1, \ldots, s_d\}$ is shattered by $\mathcal{C}$ (we now assume VC dimension $d+1$ for ease of notation). Then we can consider a distribution D that puts probability $1 - 4\varepsilon$ on s_0 and probability $4\varepsilon/d$ on each of $s_1, \ldots, s_d$.[3] For every possible labeling $(\ell_1 \ldots \ell_d) \in \{0,1\}^d$ of $s_1, \ldots, s_d$ there will be a concept $c \in \mathcal{C}$ that labels s_0 with 0, and labels s_i with ℓ_i for all $i \in \{1, \ldots, d\}$. Under D, most examples will be $(s_0, 0)$ and hence give us no information when we are learning one of those 2^d concepts. Suppose we have a learner that ε-approximates c with high probability under this D using T examples. Informally, our information-theoretic argument has the following three steps:

1. In order to ε-approximate c, the learner has to learn the c-labels of at least $3/4$ of the $s_1, \ldots, s_d$ (since together these have 4ε of the D-weight, and we want an ε-approximation). As all 2^d labelings are possible, the T examples together contain $\Omega(d)$ bits of information about c.
2. T examples give at most T times as much information about c as one example.
3. One example gives only $O(\varepsilon)$ bits of information about c, because it will tell us one of the labels of $s_1, \ldots, s_d$ only with probability 4ε (and otherwise it just gives $c(s_0) = 0$).

Putting these steps together implies $T = \Omega(d/\varepsilon)$.[4] This argument for the PAC setting is similar to an algorithmic-information argument of Apolloni and Gentile [8] and an information-theoretic argument for variants of the PAC model with noisy examples of Gentile and Helmbold [25].

[3] We remark that the distributions used here for proving lower bounds on quantum sample complexity have been used by Ehrenfeucht et al. [21] for analyzing classical PAC sample complexity.

[4] The other part of the lower bound of Eq. (1) does not depend on d and is fairly easy to prove.

As far as we know, this type of reasoning has not yet been applied to the sample complexity of *agnostic* learning. To get good lower bounds there, we consider a set of distributions D_a, indexed by d-bit string a. These distributions still have the property that if a learner gets ε-close to the minimal error, then it will have to learn $\Omega(d)$ bits of information about the distribution (i.e., about a). Hence the first step of the argument remains the same. The second step of our argument also remains the same, and the third step shows an upper bound of $O(\varepsilon^2)$ on the amount of information that the learner can get from one example. This then implies $T = \Omega(d/\varepsilon^2)$. We can also reformulate this for the case where we want the *expected* additional error of the hypothesis over the best classifier in $\mathcal{C}$ to be at most ε, which is how lower bounds are often stated in learning theory. We emphasize that our information-theoretic proof is simpler than the proofs in [7, 13, 45, 39].

This information-theoretic approach recovers the optimal classical bounds on sample complexity, but also generalizes readily to the quantum case where the learner gets T quantum examples. To obtain lower bounds on quantum sample complexity we use the same distributions D (now corresponding to a coherent quantum state) and basically just need to re-analyze the third step of the argument. In the PAC setting we show that one quantum example gives at most $O(\varepsilon \log(d/\varepsilon))$ bits of information about c, and in the agnostic setting it gives $O(\varepsilon^2 \log(d/\varepsilon))$ bits. This implies lower bounds on sample complexity that are only a logarithmic factor worse than the optimal classical bounds for the PAC setting (Eq. (1)) and the agnostic setting (Eq. (2)). This is not quite optimal yet, but already better than the previous best known lower bound (Eq. (3)). The logarithmic loss in step 3 is actually inherent in this information-theoretic argument: in some cases a quantum example *can* give roughly $\varepsilon \log d$ bits of information about c, for example when c comes from the concept class of linear functions.

1.2.2 A state-identification argument

In order to get rid of the logarithmic factor we then try another proof approach, which views learning from quantum examples as a quantum state identification problem: we are given T copies of the quantum example for some concept c and need to ε-approximate c from this. In order to render ε-approximation of c equivalent to exact identification of c, we use good linear error-correcting codes, restricting to concepts whose d-bit labeling of the elements of the shattered set $s_1, \ldots, s_d$ corresponds to a codeword. We then have $2^{\Omega(d)}$ possible concepts, one for each codeword, and need to identify the target concept from a quantum state that is the tensor product of T identical quantum examples.

State-identification problems have been well studied, and many tools are available for analyzing them. In particular, we will use the so-called "Pretty Good Measurement" (PGM, also known as "square root measurement" [29]) introduced by Hausladen and Wootters [30]. The PGM is a specific measurement that one can always use for state identification, and whose success probability is no more than quadratically worse than that of the very best measurement.[5] In Section 4 we use Fourier analysis to give an exact analysis of the average success probability of the PGM on the state-identification problems that come from both the PAC and the agnostic model. This analysis could be useful in other settings as well. Here it implies that the number of quantum examples, T, is lower bounded by Eq. (1) in the PAC setting, and by Eq. (2) in the agnostic setting.

[5] Even better, in our application the PGM *is* the optimal measurement, though this is not essential for our proof.

Using the Pretty Good Measurement, we are also able to prove lower bounds for PAC learning under *random classification noise*, which models the real-world situation that the learning data can have some errors. Classically in the random classification noise model (introduced by Angluin and Laird [6]), instead of obtaining labeled examples $(x, c(x))$ for some unknown $c \in \mathcal{C}$, the learner obtains *noisy examples* (x, b_x), where $b_x = c(x)$ with probability $1-\eta$ and $b_x = 1 - c(x)$ with probability η, for some *noise rate* $\eta \in [0, 1/2)$. Similarly, in the quantum learning model we could naturally define a *noisy quantum example* as an $(n+1)$-qubit state

$$\sum_{x \in \{0,1\}^n} \sqrt{(1-\eta)D(x)}|x, c(x)\rangle + \sqrt{\eta D(x)}|x, 1-c(x)\rangle.$$

Using the PGM, we are able to show that the quantum sample complexity of PAC learning a concept class $\mathcal{C}$ under random classification noise is:

$$\Omega\Big(\frac{d}{(1-2\eta)^2\varepsilon} + \frac{\log(1/\delta)}{(1-2\eta)^2\varepsilon}\Big). \tag{4}$$

We remark here that the best known classical sample complexity lower bound (see [46]) under the random classification noise is equal to the quantum sample complexity lower bound proven in Eq. (4).

1.3 Related work

Let us briefly discuss some related work in quantum learning theory, referring to our recent survey [9] for more. In this paper we focus on *sample* complexity, which is a fundamental information-theoretic quantity. Sample complexity concerns a form of "passive" learning: the learner gets a number of examples at the start of the process, and then has to extract enough information about the target concept from these. We may also consider more active learning settings, in particular ones where the learner can make membership queries (i.e., learn the label $c(x)$ for any x of his choice). Servedio and Gortler [44] showed that in this setting, classical and quantum complexity are polynomially related. They also exhibit an example of a factor-n speed-up from quantum membership queries using the Bernstein-Vazirani algorithm. Jackson et al. [34] showed how quantum membership queries can improve Jackson's classical algorithm for learning DNF with membership queries under the uniform distribution [33].

For *quantum exact learning* (also referred to as the *oracle identification* problem in the quantum literature), Kothari [40] resolved a conjecture of Hunziker et al. [32], that states that for any concept class $\mathcal{C}$, the number of quantum membership queries required to exactly identify a concept $c \in \mathcal{C}$ is $O(\frac{\log|\mathcal{C}|}{\sqrt{\hat{\gamma}^{\mathcal{C}}}})$, where $\hat{\gamma}^{\mathcal{C}}$ is a combinatorial parameter of the concept class $\mathcal{C}$ which we shall not define here (see [10] for a precise definition). Montanaro [42] showed how low-degree polynomials over a finite field can be identified more efficiently using quantum algorithms.

In many ways the *time* complexity of learning is at least as important as the sample complexity. We already mentioned that Servedio and Gortler [44] exhibited a concept class based on factoring Blum integers that can be learned in quantum polynomial time but not in classical polynomial time, unless Blum integers can be factored efficiently. Under the weaker (but still widely believed) assumption that one-way functions exist, they exhibited a concept class that can be learned exactly in polynomial time using quantum membership queries, but that takes superpolynomial time to learn from classical membership queries. Gavinsky [24] introduced a model of learning called "Predictive Quantum" (PQ), a variation of quantum

PAC learning, and exhibited a *relational* concept class that is polynomial-time learnable in PQ, while any "reasonable" classical model requires an exponential number of classical examples to learn the concept class.

Aïmeur et al. [3, 4] consider a number of quantum algorithms in learning contexts such as clustering via minimum spanning tree, divisive clustering, and k-medians, using variants of Grover's algorithm [26] to improve the time complexity of the analogous classical algorithms. Recently, there have been some quantum machine learning algorithms based on the HHL algorithm [28] for solving (in a weak sense) very well-behaved linear systems. However, these algorithms often come with some fine print that limits their applicability, and their advantage over classical is not always clear. We refer to Aaronson [2] for references and caveats. There has also been some work on quantum training of neural networks [53, 54].

In addition to learning classical objects such as Boolean functions, one may also study the learnability of *quantum* objects. In particular, Aaronson [1] studied how well n-qubit quantum states can be learned from measurement results. In general, an n-qubit state ρ is specified by $\exp(n)$ many parameters, and $\exp(n)$ measurement results on equally many copies of ρ are needed to learn a good approximation of ρ (say, in trace distance). However, Aaronson showed an interesting and surprisingly efficient PAC-like result: from $O(n)$ measurement results, with measurements chosen i.i.d. according to an unknown distribution D on the set of all possible two-outcome measurements, we can learn an n-qubit quantum state $\widetilde{\rho}$ that has roughly the same expectation value as ρ for "most" possible two-outcome measurements. In the latter, "most" is again measured under D, just like in the usual PAC learning the error of the learner's hypothesis is evaluated under the same distribution D that generated the learner's examples. Accordingly, $O(n)$ rather than $\exp(n)$ measurement results suffice to approximately learn an n-qubit state for most practical purposes.

The use of Fourier analysis in analyzing the success probability of the Pretty Good Measurement in quantum state identification appears in a number of earlier works. By considering the dihedral hidden subgroup problem (DHSP) as a state identification problem, Bacon et al. [14] show that the PGM is the optimal measurement for DHSP and prove a lower bound on the sample complexity of $\Omega(\log|\mathcal{G}|)$ for a dihedral group $\mathcal{G}$ using Fourier analysis. Ambainis and Montanaro [5] view the "search with wildcard" problem as a state identification problem. Using ideas similar to ours, they show that the (x, y)-th entry of the Gram matrix for the ensemble depends on the Hamming distance between x and y, allowing them to use Fourier analysis to obtain an upper bound on the success probability of the state identification problem using the PGM.

1.4 Organization

In Section 2 we formally define the classical and quantum learning models and introduce the Pretty Good Measurement. In Section 3 we prove our information-theoretic lower bounds both for classical and quantum learning. In Section 4 we prove an optimal quantum lower bound for PAC and agnostic learning by viewing the learning process as a state identification problem. We conclude in Section 5 with some open questions for further work.

2 Preliminaries

2.1 Notation

Let $[n] = \{1, \ldots, n\}$. For $x, y \in \{0,1\}^d$, the bit-wise sum $x + y$ is over $\mathbb{F}_2$, the *Hamming distance* $d(x, y)$ is the number of indices on which x and y differ, $|x+y|$ is the Hamming weight

of the string $x+y$ (which equals $d_H(x,y)$), and $x \cdot y = \sum_i x_i y_i$ (where the sum is over $\mathbb{F}_2$). For an n-dimensional vector space, the standard basis is denoted by $\{e_i \in \{0,1\}^n : i \in [n]\}$, where e_i is the vector with a 1 in the i-th coordinate and 0's elsewhere. We write log for logarithm to base 2, and ln for base e. We will often use the bijection between the sets $\{0,1\}^k$ and $[2^k]$ throughout this paper. Let $1_{[A]}$ be the indicator for an event A, and let $\delta_{x,y} = 1_{[x=y]}$. We denote random variables in bold, such as $\mathbf{A}$, $\mathbf{B}$.

For a Boolean function $f : \{0,1\}^m \to \{0,1\}$ and $M \in \mathbb{F}_2^{m \times k}$ we define $f \circ M : \{0,1\}^k \to \{0,1\}$ as $(f \circ M)(x) := f(Mx)$ (where the matrix-vector product is over $\mathbb{F}_2$) for all $x \in \{0,1\}^k$. For a distribution $D : \{0,1\}^n \to [0,1]$, let $\mathrm{supp}(D) = \{x \in \{0,1\}^n : D(x) \neq 0\}$. By $x \sim D$, we mean x is sampled according to the distribution D, i.e., $\Pr[\mathbf{X} = x] = D(x)$.

If M is a positive semidefinite (psd) matrix, we define $\sqrt{M}$ as the unique psd matrix that satisfies $\sqrt{M} \cdot \sqrt{M} = M$, and $\sqrt{M}(i,j)$ as the (i,j)-th entry of $\sqrt{M}$. For a matrix $A \in \mathbb{R}^{m \times n}$, we denote the singular values of A by $\sigma_1(A) \geq \sigma_2(A) \geq \cdots \geq \sigma_{\min\{m,n\}}(A) \geq 0$. The spectral norm of A is $\|A\| = \max_{x \in \mathbb{R}^n, \|x\|=1} \|Ax\| = \sigma_1$. Given a set of d-dimensional vectors $U = \{u_1, \ldots, u_n\} \in \mathbb{R}^d$, the Gram matrix V corresponding to the set U is the $n \times n$ psd matrix defined as $V(i,j) = u_i^t u_j$ for $i,j \in [n]$, where u_i^t is the row vector that is the transpose of the column vector u_i.

A technical tool used in our analysis of state identification problems is Fourier analysis on the Boolean cube. We will just introduce the basics of Fourier analysis here, referring to [43] for more. Define the inner product between functions $f, g : \{0,1\}^n \to \mathbb{R}$ as

$$\langle f, g \rangle = \mathop{\mathbb{E}}_x [f(x) \cdot g(x)]$$

where the expectation is uniform over $x \in \{0,1\}^n$. For $S \subseteq [n]$ (equivalently $S \in \{0,1\}^n$), let $\chi_S(x) := (-1)^{S \cdot x}$ denote the parity of the variables (of x) indexed by the set S. It is easy to see that the set of functions $\{\chi_S\}_{S \subseteq [n]}$ forms an orthonormal basis for the space of real-valued functions over the Boolean cube. Hence every f can be decomposed as

$$f(x) = \sum_{S \subseteq [n]} \widehat{f}(S)(-1)^{S \cdot x} \qquad \text{for all } x \in \{0,1\}^n,$$

where $\widehat{f}(S) = \langle f, \chi_S \rangle = \mathbb{E}_x[f(x) \cdot \chi_S(x)]$ is called a *Fourier coefficient* of f.

2.2 Learning in general

In machine learning, a concept class $\mathcal{C}$ over $\{0,1\}^n$ is a set of concepts $c : \{0,1\}^n \to \{0,1\}$. We refer to a concept class $\mathcal{C}$ as being *trivial* if either $\mathcal{C}$ contains only one concept, or $\mathcal{C}$ contains two concepts c_0, c_1 with $c_0(x) = 1 - c_1(x)$ for every $x \in \{0,1\}^n$. For $c : \{0,1\}^n \to \{0,1\}$, we will often refer to the tuple $(x, c(x)) \in \{0,1\}^{n+1}$ as a *labeled example*, where $c(x)$ is the *label* of x.

A central combinatorial concept in learning is the Vapnik-Chervonenkis (VC) dimension [50]. Fix a concept class $\mathcal{C}$ over $\{0,1\}^n$. A set $\mathcal{S} = \{s_1, \ldots, s_t\} \subseteq \{0,1\}^n$ is said to be *shattered* by a concept class $\mathcal{C}$ if $\{(c(s_1), \ldots, c(s_t)) : c \in \mathcal{C}\} = \{0,1\}^t$. In other words, for every labeling $\ell \in \{0,1\}^t$, there exists a $c \in \mathcal{C}$ such that $(c(s_1), \ldots, c(s_t)) = \ell$. The VC dimension of a concept class $\mathcal{C}$ is the size of the largest $\mathcal{S} \subseteq \{0,1\}^n$ that is shattered by $\mathcal{C}$.

2.3 Classical learning models

In this paper we will be concerned mainly with the PAC (Probably Approximately Correct) model of learning introduced by Valiant [49], and the agnostic model of learning introduced

by Haussler [31] and Kearns et al. [36]. For further reading, see standard textbooks in computational learning theory such as [38, 7, 45].

In the classical PAC model, a learner $\mathcal{A}$ is given access to a *random example oracle* $\mathrm{PEX}(c, D)$ which generates labeled examples of the form $(x, c(x))$ where x is drawn from an unknown distribution $D : \{0,1\}^n \to [0,1]$ and $c \in \mathcal{C}$ is the *target concept* that $\mathcal{A}$ is trying to learn. For a concept $c \in \mathcal{C}$ and hypothesis $h : \{0,1\}^n \to \{0,1\}$, we define the error of h compared to the target concept c, under D, as $\mathrm{err}_D(h, c) = \Pr_{x\sim D}[h(x) \neq c(x)]$. A learning algorithm $\mathcal{A}$ is an (ε, δ)-*PAC learner* for $\mathcal{C}$, if the following holds:

For every $c \in \mathcal{C}$ and distribution D, given access to the $\mathrm{PEX}(c, D)$ oracle:
$\mathcal{A}$ outputs an h such that $\mathrm{err}_D(h, c) \leq \varepsilon$ with probability at least $1 - \delta$.

The *sample complexity* of $\mathcal{A}$ is the maximum number of invocations of the $\mathrm{PEX}(c, D)$ oracle which the learner makes, over all concepts $c \in \mathcal{C}$, distributions D, and the internal randomness of the learner. The (ε, δ)-*PAC sample complexity* of a concept class $\mathcal{C}$ is the minimum sample complexity over all (ε, δ)-PAC learners for $\mathcal{C}$.

Agnostic learning is the following model: for a distribution $D : \{0,1\}^{n+1} \to [0,1]$, a learner $\mathcal{A}$ is given access to an $\mathrm{AEX}(D)$ oracle that generates examples of the form (x, b) drawn from the distribution D. We define the error of $h : \{0,1\}^n \to \{0,1\}$ under D as $\mathrm{err}_D(h) = \Pr_{(x,b)\sim D}[h(x) \neq b]$. When h is restricted to come from a concept class $\mathcal{C}$, the minimal error achievable is $\mathrm{opt}_D(\mathcal{C}) = \min_{c\in\mathcal{C}}\{\mathrm{err}_D(c)\}$. In agnostic learning, a learner $\mathcal{A}$ needs to output a hypothesis h whose error is not much bigger than $\mathrm{opt}_D(\mathcal{C})$. A learning algorithm $\mathcal{A}$ is an (ε, δ)-*agnostic learner* for $\mathcal{C}$ if:

For every distribution D on $\{0,1\}^{n+1}$, given access to the $\mathrm{AEX}(D)$ oracle:
$\mathcal{A}$ outputs an $h \in \mathcal{C}$ such that $\mathrm{err}_D(h) \leq \mathrm{opt}_D(\mathcal{C}) + \varepsilon$ with probability at least $1 - \delta$.

Note that if there is a $c \in \mathcal{C}$ which perfectly classifies every x with label y for $(x, y) \in \mathrm{supp}(D)$, then $\mathrm{opt}_D(\mathcal{C}) = 0$ and we are in the setting of proper PAC learning. The *sample complexity* of $\mathcal{A}$ is the maximum number of invocations of the $\mathrm{AEX}(c, D)$ oracle which the learner makes, over all distributions D and over the learner's internal randomness. The (ε, δ)-*agnostic sample complexity* of a concept class $\mathcal{C}$ is the minimum sample complexity over all (ε, δ)-agnostic learners for $\mathcal{C}$.

2.4 Quantum information theory

Throughout this paper we will assume the reader is familiar with the following quantum terminology. An n-dimensional *pure state* is $|\psi\rangle = \sum_{i=1}^n \alpha_i |i\rangle$, where $|i\rangle$ is the n-dimensional unit vector that has a 1 only at position i, the α_i's are complex numbers called the *amplitudes*, and $\sum_{i\in[n]} |\alpha_i|^2 = 1$. An n-dimensional *mixed state* (or *density matrix*) $\rho = \sum_{i=1}^n p_i |\psi_i\rangle\langle\psi_i|$ is a mixture of pure states $|\psi_1\rangle, \ldots, |\psi_n\rangle$ prepared with probabilities $p_1, \ldots, p_n$, respectively. The eigenvalues $\lambda_1, \ldots, \lambda_n$ of ρ are non-negative reals and satisfy $\sum_{i\in[n]} \lambda_i = 1$. If ρ is pure (i.e., $\rho = |\psi\rangle\langle\psi|$ for some $|\psi\rangle$), then one of the eigenvalues is 1 and the others are 0.

To obtain classical information from ρ, one could apply a POVM (positive-operator-valued measure) to the state ρ. An m-outcome POVM is specified by a set of positive semidefinite matrices $\{M_i\}_{i\in[m]}$ with the property $\sum_i M_i = \mathrm{Id}$. When this POVM is applied to the mixed state ρ, the probability of the j-th outcome is given by $\mathrm{Tr}(M_j \rho)$.

For a probability vector $(p_1, \ldots, p_k)$ (where $\sum_{i\in[k]} p_i = 1$), the entropy function is defined as $H(p_1, \ldots, p_k) = -\sum_{i\in[k]} p_i \log p_i$. When $k = 2$, with $p_1 = p$ and $p_2 = 1 - p$, we denote the binary entropy function as $H(p)$. For a state ρ_{AB} on the Hilbert space $\mathcal{H}_A \otimes \mathcal{H}_B$, we let ρ_A be the reduced state after taking the partial trace over $\mathcal{H}_B$. The

entropy of a quantum state ρ_A is defined as $S(\mathbf{A}) = -\mathrm{Tr}(\rho_A \log \rho_A)$. The mutual information is defined as $I(\mathbf{A} : \mathbf{B}) = S(\mathbf{A}) + S(\mathbf{B}) - S(\mathbf{AB})$, and conditional entropy is defined as $S(\mathbf{A}|\mathbf{B}) = S(\mathbf{AB}) - S(\mathbf{B})$. Classical information-theoretic quantities correspond to the special case where ρ is a diagonal matrix whose diagonal corresponds to the probability distribution of the random variable. Writing ρ_A in its eigenbasis, it follows that $S(\mathbf{A}) = H(\lambda_1, \ldots, \lambda_{\dim(\rho_A)})$, where $\lambda_1, \ldots, \lambda_{\dim(\rho_A)}$ are the eigenvalues of ρ. If ρ_A is a pure state, $S(\mathbf{A}) = 0$.

2.5 Quantum learning models

The quantum PAC learning model was introduced by Bshouty and Jackson in [18]. The quantum PAC model is a generalization of the classical PAC model, instead of having access to random examples $(x, c(x))$ from the $\mathrm{PEX}(c, D)$ oracle, the learner now has access to superpositions over all $(x, c(x))$. For an unknown distribution $D : \{0,1\}^n \to [0,1]$ and concept $c \in \mathcal{C}$, a *quantum example oracle* $\mathrm{QPEX}(c, D)$ acts on $|0^n, 0\rangle$ and produces a *quantum example* $\sum_{x \in \{0,1\}^n} \sqrt{D(x)}|x, c(x)\rangle$ (we leave QPEX undefined on other basis states). A quantum learner is given access to some copies of the state generated by $\mathrm{QPEX}(c, D)$ and performs a POVM where each outcome is associated with a hypothesis. A learning algorithm $\mathcal{A}$ is an (ε, δ)-*PAC quantum learner* for $\mathcal{C}$ if:

> For every $c \in \mathcal{C}$ and distribution D, given access to the $\mathrm{QPEX}(c, D)$ oracle:
> $\mathcal{A}$ outputs an h such that $\mathrm{err}_D(h, c) \leq \varepsilon$, with probability at least $1 - \delta$.

The *sample complexity* of $\mathcal{A}$ is the maximum number invocations of the $\mathrm{QPEX}(c, D)$ oracle, maximized over all $c \in \mathcal{C}$, distributions D, and the learner's internal randomness. The (ε, δ)-*PAC quantum sample complexity* of a concept class $\mathcal{C}$ is the minimum sample complexity over all (ε, δ)-PAC quantum learners for $\mathcal{C}$.

We define quantum agnostic learning now. For a joint distribution $D : \{0,1\}^{n+1} \to [0,1]$ over the set of examples, the learner has access to an $\mathrm{QAEX}(D)$ oracle which acts on $|0^n, 0\rangle$ and produces a quantum example $\sum_{(x,b) \in \{0,1\}^{n+1}} \sqrt{D(x,b)}|x, b\rangle$. A learning algorithm $\mathcal{A}$ is an (ε, δ)-*agnostic quantum learner* for $\mathcal{C}$ if:

> For every distribution D, given access to the $\mathrm{QAEX}(D)$ oracle:
> $\mathcal{A}$ outputs an $h \in \mathcal{C}$ such that $\mathrm{err}_D(h) \leq \mathrm{opt}_D(\mathcal{C}) + \varepsilon$ with probability at least $1 - \delta$.

The *sample complexity* of $\mathcal{A}$ is the maximum number invocations of the $\mathrm{QAEX}(D)$ oracle over all distributions D and over the learner's internal randomness. The (ε, δ)-*agnostic quantum sample complexity* of a concept class $\mathcal{C}$ is the minimum sample complexity over all (ε, δ)-agnostic quantum learners for $\mathcal{C}$.

2.6 Pretty Good Measurement

Consider an ensemble of d-dimensional states, $\mathcal{E} = \{(p_i, |\psi_i\rangle)\}_{i \in [m]}$, where $\sum_{i \in [m]} p_i = 1$. Suppose we are given an unknown state $|\psi_i\rangle$ sampled according to the probabilities and we are interested in maximizing the average probability of success to identify the state that we are given. For a POVM specified by positive semidefinite matrices $\mathcal{M} = \{M_i\}_{i \in [m]}$, the probability of obtaining outcome j equals $\langle \psi_i | M_j | \psi_i \rangle$. The average success probability is defined as

$$P_{\mathcal{M}}(\mathcal{E}) = \sum_{i=1}^{m} p_i \langle \psi_i | M_i | \psi_i \rangle.$$

Let $P^{opt}(\mathcal{E}) = \max_{\mathcal{M}} P_{\mathcal{M}}(\mathcal{E})$ denote the optimal average success probability of $\mathcal{E}$, where the maximization is over the set of valid m-outcome POVMs.

For every ensemble $\mathcal{E}$, the so-called *Pretty Good Measurement* (PGM) is a specific POVM (depending on the ensemble $\mathcal{E}$), which we shall define shortly, that does *reasonably* well against $\mathcal{E}$. Suppose $P^{PGM}(\mathcal{E})$ is defined as the average success probability of identifying the states in $\mathcal{E}$ using the PGM, then we have that

$$P^{opt}(\mathcal{E})^2 \leq P^{PGM}(\mathcal{E}) \leq P^{opt}(\mathcal{E}),$$

where the second inequality follows because $P^{opt}(\mathcal{E})$ is a maximization over all valid POVMs and the first inequality was shown by Barnum and Knill [15].

For completeness we give a simple proof of $P^{opt}(\mathcal{E})^2 \leq P^{PGM}(\mathcal{E})$ below (similar to [41]). Let $|\psi'_i\rangle = \sqrt{p_i}|\psi_i\rangle$, and $\mathcal{E}' = \{|\psi'_i\rangle : i \in [m]\}$ be the set of states in $\mathcal{E}$, renormalized to reflect their probabilities. Define $\rho = \sum_{i\in[m]} |\psi'_i\rangle\langle\psi'_i|$. The PGM is defined as the set of measurement operators $\{|\nu_i\rangle\langle\nu_i|\}_{i\in[m]}$ where $|\nu_i\rangle = \rho^{-1/2}|\psi'_i\rangle$ (the inverse square root of ρ is taken over its non-zero eigenvalues). We first verify this is a valid POVM:

$$\sum_{i=1}^{m} |\nu_i\rangle\langle\nu_i| = \rho^{-1/2}\left(\sum_{i=1}^{m} |\psi'_i\rangle\langle\psi'_i|\right)\rho^{-1/2} = \mathrm{Id}\,.$$

Let G be the Gram matrix for the set $\mathcal{E}'$, i.e., $G(i,j) = \langle\psi'_i|\psi'_j\rangle$ for $i,j \in [m]$. It can be verified that $\sqrt{G}(i,j) = \langle\psi'_i|\rho^{-1/2}|\psi'_j\rangle$. Hence

$$P^{PGM}(\mathcal{E}) = \sum_{i\in[m]} p_i|\langle\nu_i|\psi_i\rangle|^2 = \sum_{i\in[m]} |\langle\nu_i|\psi'_i\rangle|^2 = \sum_{i\in[m]} \langle\psi'_i|\rho^{-1/2}|\psi'_i\rangle^2 = \sum_{i\in[m]} \sqrt{G}(i,i)^2.$$

We now prove $P^{opt}(\mathcal{E})^2 \leq P^{PGM}(\mathcal{E})$. Suppose $\mathcal{M}$ is the optimal measurement. Since $\mathcal{E}$ consists of pure states, by a result of Eldar et al. [23], we can assume without loss of generality that the measurement operators in $\mathcal{M}$ are rank-1, so $M_i = |\mu_i\rangle\langle\mu_i|$ for some $|\mu_i\rangle$. Note that

$$\begin{aligned} 1 = \mathrm{Tr}(\rho) &= \mathrm{Tr}\Big(\sum_{i\in[m]} |\mu_i\rangle\langle\mu_i|\rho^{1/2} \sum_{j\in[m]} |\mu_j\rangle\langle\mu_j|\rho^{1/2}\Big) \\ &= \sum_{i,j\in[m]} |\langle\mu_i|\rho^{1/2}|\mu_j\rangle|^2 \geq \sum_{i\in[m]} \langle\mu_i|\rho^{1/2}|\mu_i\rangle^2. \end{aligned} \tag{5}$$

Then, using the Cauchy-Schwarz inequality, we have

$$\begin{aligned} P^{opt}(\mathcal{E}) = \sum_{i\in[m]} |\langle\mu_i|\psi'_i\rangle|^2 &= \sum_{i\in[m]} |\langle\mu_i|\rho^{1/4}\rho^{-1/4}|\psi'_i\rangle|^2 \\ &\leq \sum_{i\in[m]} \langle\mu_i|\rho^{1/2}|\mu_i\rangle\langle\psi'_i|\rho^{-1/2}|\psi'_i\rangle \\ &\leq \sqrt{\sum_{i\in[m]} \langle\mu_i|\rho^{1/2}|\mu_i\rangle^2}\sqrt{\sum_{i\in[m]} \langle\psi'_i|\rho^{-1/2}|\psi'_i\rangle^2} \\ &\overset{\text{Eq. (5)}}{\leq} \sqrt{\sum_{i\in[m]} \langle\psi'_i|\rho^{-1/2}|\psi'_i\rangle^2} = \sqrt{P^{PGM}(\mathcal{E})}. \end{aligned}$$

The above shows that for all ensembles $\mathcal{E}$, the PGM for that ensemble is not much worse than the optimal measurement. In some cases the PGM *is* the optimal measurement. In particular, an ensemble $\mathcal{E}$ is called *geometrically uniform* if $\mathcal{E} = \{U_i|\varphi\rangle : i \in [m]\}$ for some Abelian group of matrices $\{U_i\}_{i\in[m]}$ and state $|\varphi\rangle$. Eldar and Forney [22] showed $P^{opt}(\mathcal{E}) = P^{PGM}(\mathcal{E})$ for such $\mathcal{E}$.

2.7 Known results and required claims

The following theorems characterize the sample complexity of classical PAC and agnostic learning.

▶ **Theorem 1** ([17, 27]). *Let $\mathcal{C}$ be a concept class with VC-dim$(\mathcal{C}) = d+1$. In the PAC model, $\Theta\Big(\frac{d}{\varepsilon} + \frac{\log(1/\delta)}{\varepsilon}\Big)$ examples are necessary and sufficient for a classical (ε, δ)-PAC learner for $\mathcal{C}$.*

▶ **Theorem 2** ([51, 46, 48]). *Let $\mathcal{C}$ be a concept class with VC-dim$(\mathcal{C}) = d$. In the agnostic model, $\Theta\Big(\frac{d}{\varepsilon^2} + \frac{\log(1/\delta)}{\varepsilon^2}\Big)$ examples are necessary and sufficient for a classical (ε, δ)-agnostic learner for $\mathcal{C}$.*

We will use the following well-known theorem from the theory of error-correcting codes:

▶ **Theorem 3.** *For every sufficiently large integer n, there exists an integer $k \in [n/4, n]$ and a matrix $M \in \mathbb{F}_2^{n\times k}$ of rank k, such that the associated $[n, k, d]_2$ linear code $\{Mx : x \in \{0,1\}^k\}$ has minimal distance $d \geq n/8$.*

We will need the following claims later

▶ **Claim 4.** *Let $f : \{0,1\}^m \to \mathbb{R}$ and let $M \in \mathbb{F}_2^{m\times k}$. Then the Fourier coefficients of $f \circ M$ are $\widehat{f \circ M}(Q) = \sum_{S\in\{0,1\}^m : M^tS=Q} \widehat{f}(S)$ for all $Q \subseteq [k]$ (where M^t is the transpose of the matrix M).*

Proof. Writing out the Fourier coefficients of $f \circ M$

$$\begin{aligned}
\widehat{f \circ M}(Q) &= \mathop{\mathbb{E}}_{z\in\{0,1\}^k}[(f \circ M)(z)(-1)^{Q\cdot z}] \\
&= \mathop{\mathbb{E}}_{z\in\{0,1\}^k}\Big[\sum_{S\in\{0,1\}^m} \widehat{f}(S)(-1)^{S\cdot(Mz)+Q\cdot z}\Big] && \text{(Fourier expansion of } f\text{)} \\
&= \sum_{S\in\{0,1\}^m} \widehat{f}(S) \mathop{\mathbb{E}}_{z\in\{0,1\}^k}[(-1)^{(M^tS+Q)\cdot z}] && \text{(using } \langle S, Mz\rangle = \langle M^tS, z\rangle\text{)} \\
&= \sum_{S:M^tS=Q} \widehat{f}(S). && \text{(using } \mathbb{E}_{z\in\{0,1\}^k}(-1)^{(z_1+z_2)\cdot z} = \delta_{z_1,z_2}\text{)}
\end{aligned}$$

◀

▶ **Claim 5.** $\max\{(c/\sqrt{t})^t : t \in [1, c^2]\} = e^{c^2/(2e)}$.

Proof. The value of t at which the function $\Big(c/\sqrt{t}\Big)^t$ is the largest, is obtained by differentiating the function with respect to t,

$$\frac{d}{dt}\Big(c/\sqrt{t}\Big)^t = (c/\sqrt{t})^t\Big(\ln(c/\sqrt{t}) - 1/2\Big).$$

Equating the derivative to zero we obtain the maxima (the second derivative can be checked to be negative) at $t = c^2/e$. ◀

▶ **Fact 6.** *For all $\varepsilon \in [0, 1/2]$ we have $H(\varepsilon) \leq O(\varepsilon \log(1/\varepsilon))$, and (from the Taylor series)*

$$1 - H(1/2 + \varepsilon) \leq 2\varepsilon^2/\ln 2 + O(\varepsilon^4).$$

▶ **Fact 7.** *For every positive integer n, we have that $\binom{n}{k} \leq 2^{nH(k/n)}$ for all $k \leq n$ and $\sum_{i=0}^{m}\binom{n}{i} \leq 2^{nH(m/n)}$ for all $m \leq n/2$.*

The following facts are well-known in quantum information theory.

▶ **Fact 8.** *Let binary random variable* $\mathbf{b} \in \{0,1\}$ *be uniformly distributed. Suppose an algorithm is given* $|\psi_{\mathbf{b}}\rangle$ *(for unknown* b*) and is required to guess whether* $\mathbf{b}=0$ *or* $\mathbf{b}=1$*. It will guess correctly with probability at most* $\frac{1}{2}+\frac{1}{2}\sqrt{1-|\langle\psi_0|\psi_1\rangle|^2}$*.*

Note that if we can distinguish $|\psi_0\rangle$ and $|\psi_1\rangle$ with probability $\geq 1-\delta$, then $|\langle\psi_0|\psi_1\rangle| \leq 2\sqrt{\delta(1-\delta)}$.

▶ **Fact 9.** *(Subadditivity of quantum entropy): For an arbitrary bipartite state* ρ_{AB} *on the Hilbert space* $\mathcal{H}_A \otimes \mathcal{H}_B$*, it holds that* $S(\rho_{AB}) \leq S(\rho_A) + S(\rho_B)$*.*

3 Information-theoretic lower bounds

Upper bounds on sample complexity carry over from classical to quantum PAC learning, because a quantum example becomes a classical example if we just measure it. Our main goal is to show that the *lower* bounds also carry over. All our lower bounds will involve two terms, one that is independent of $\mathcal{C}$ and one that is dependent on the VC dimension of $\mathcal{C}$. In Section 3.1 we prove the VC-independent part of the lower bounds for the *quantum* setting (which also is a lower bound for the classical setting), in Section 3.2 we present an information-theoretic lower bound on sample complexity for PAC learning and agnostic learning which yields optimal VC-dependent bounds in the classical case. Using similar ideas, in Section 3.3 we obtain near-optimal bounds in the quantum case.

3.1 VC-independent part of lower bounds

▶ **Lemma 10** ([10])**.** *Let* $\mathcal{C}$ *be a non-trivial concept class. For every* $\delta \in (0,1/2)$*,* $\varepsilon \in (0,1/4)$*, a* (ε,δ)*-PAC quantum learner for* $\mathcal{C}$ *has sample complexity* $\Omega(\frac{1}{\varepsilon}\log\frac{1}{\delta})$*.*

Proof. Since $\mathcal{C}$ is non-trivial, we may assume there are two concepts $c_1, c_2 \in \mathcal{C}$ defined on two inputs $\{x_1, x_2\}$ as follows $c_1(x_1) = c_2(x_1) = 0$ and $c_1(x_2) = 0, c_2(x_2) = 1$. Consider the distribution $D(x_1) = 1-\varepsilon$ and $D(x_2) = \varepsilon$. For $i \in \{1,2\}$, the state of the algorithm after T queries to $\mathrm{QPEX}(c_i, D)$ is $|\psi_i\rangle = (\sqrt{1-\varepsilon}|x_1, 0\rangle + \sqrt{\varepsilon}|x_2, c_i(x_2)\rangle)^{\otimes T}$. It follows that $\langle\psi_1|\psi_2\rangle = (1-\varepsilon)^T$. Since the success probability of an (ε,δ)-PAC quantum learner is $\geq 1-\delta$, Fact 8 implies $\langle\psi_1|\psi_2\rangle \leq 2\sqrt{\delta(1-\delta)}$. Hence $T = \Omega(\frac{1}{\varepsilon}\log\frac{1}{\delta})$. ◀

▶ **Lemma 11.** *Let* $\mathcal{C}$ *be a non-trivial concept class. For every* $\delta \in (0,1/2)$*,* $\varepsilon \in (0,1/4)$*, a* (ε,δ)*-agnostic quantum learner for* $\mathcal{C}$ *has sample complexity* $\Omega(\frac{1}{\varepsilon^2}\log\frac{1}{\delta})$*.*

Proof. Since $\mathcal{C}$ is non-trivial, we may assume there are two concepts $c_1, c_2 \in \mathcal{C}$ and there exists an input $x \in \{0,1\}^n$ such that $c_1(x) \neq c_2(x)$. Consider the two distributions D_- and D_+ defined as follows: $D_\pm(x, c_1(x)) = (1\pm\varepsilon)/2$ and $D_\pm(x, c_2(x)) = (1\mp\varepsilon)/2$. Let $|\psi_\pm\rangle$ be the state after T queries to $\mathrm{QAEX}(D_\pm)$, i.e., $|\psi_\pm\rangle = (\sqrt{(1\pm\varepsilon)/2}|x, c_1(x)\rangle + \sqrt{(1\mp\varepsilon)/2}|x, c_2(x)\rangle)^{\otimes T}$. It follows that $\langle\psi_+|\psi_-\rangle = (1-\varepsilon^2)^{T/2}$. Since the success probability of an (ε,δ)-agnostic quantum learner is $\geq 1-\delta$, Fact 8 implies $\langle\psi_+|\psi_-\rangle \leq 2\sqrt{\delta(1-\delta)}$. Hence $T = \Omega(\frac{1}{\varepsilon^2}\log\frac{1}{\delta})$ ◀

3.2 Information-theoretic lower bounds on sample complexity: classical case

3.2.1 Optimal lower bound for classical PAC learning

▶ **Theorem 12.** *Let $\mathcal{C}$ be a concept class with VC-dim$(\mathcal{C}) = d+1$. Then for every $\delta \in (0, 1/2)$ and $\varepsilon \in (0, 1/4)$, every (ε, δ)-PAC learner for $\mathcal{C}$ has sample complexity $\Omega\left(\frac{d}{\varepsilon} + \frac{\log(1/\delta)}{\varepsilon}\right)$.*

Proof. Consider an (ε, δ)-PAC learner for $\mathcal{C}$ that uses T examples. The d-independent part of the lower bound, $T = \Omega(\log(1/\delta)/\varepsilon)$, even holds for quantum examples and was proven in Lemma 10. Hence it remains to prove $T = \Omega(d/\varepsilon)$. It suffices to show this for a specific distribution D, defined as follows. Let $\mathcal{S} = \{s_0, s_1, \ldots, s_d\} \subseteq \{0,1\}^n$ be some $(d+1)$-element set shattered by $\mathcal{C}$. Define $D(s_0) = 1 - 4\varepsilon$ and $D(s_i) = 4\varepsilon/d$ for all $i \in [d]$.

Because $\mathcal{S}$ is shattered by $\mathcal{C}$, for each string $a \in \{0,1\}^d$, there exists a concept $c_a \in \mathcal{C}$ such that $c_a(s_0) = 0$ and $c_a(s_i) = a_i$ for all $i \in [d]$. We define two correlated random variables $\mathbf{A}$ and $\mathbf{B}$ corresponding to the concept and to the examples, respectively. Let $\mathbf{A}$ be a random variable that is uniformly distributed over $\{0,1\}^d$; if $\mathbf{A} = a$, let $\mathbf{B} = \mathbf{B}_1 \ldots \mathbf{B}_T$ be T i.i.d. examples from c_a according to D. We give the following three-step analysis of these random variables:

1. $I(\mathbf{A} : \mathbf{B}) \geq (1-\delta)(1 - H(1/4))d - H(\delta) = \Omega(d)$.

   Proof. Let random variable $h(\mathbf{B}) \in \{0,1\}^d$ be the hypothesis that the learner produces (given the examples in $\mathbf{B}$) restricted to the elements $s_1, \ldots, s_d$. Note that the error of the hypothesis $\mathrm{err}_D(h(\mathbf{B}), c_{\mathbf{A}})$ equals $d_H(\mathbf{A}, h(\mathbf{B})) \cdot 4\varepsilon/d$, because each s_i where $\mathbf{A}$ and $h(\mathbf{B})$ differ contributes $D(s_i) = 4\varepsilon/d$ to the error. Let $\mathbf{Z}$ be the indicator random variable for the event that the error is $\leq \varepsilon$. If $\mathbf{Z} = 1$, then $d_H(\mathbf{A}, h(\mathbf{B})) \leq d/4$. Since we are analyzing an (ε, δ)-PAC learner, we have $\Pr[\mathbf{Z} = 1] \geq 1 - \delta$, and $H(\mathbf{Z}) \leq H(\delta)$. Given a string $h(\mathbf{B})$ that is $d/4$-close to $\mathbf{A}$, $\mathbf{A}$ ranges over a set of only $\sum_{i=0}^{d/4} \binom{d}{i} \leq 2^{H(1/4)d}$ possible d-bit strings (using Fact 7), hence $H(\mathbf{A} \mid \mathbf{B}, \mathbf{Z} = 1) \leq H(\mathbf{A} \mid h(\mathbf{B}), \mathbf{Z} = 1) \leq H(1/4)d$. We now lower bound $I(\mathbf{A} : \mathbf{B})$ as follows:

$$\begin{aligned} I(\mathbf{A} : \mathbf{B}) &= H(\mathbf{A}) - H(\mathbf{A} \mid \mathbf{B}) \\ &\geq H(\mathbf{A}) - H(\mathbf{A} \mid \mathbf{B}, \mathbf{Z}) - H(\mathbf{Z}) \\ &= H(\mathbf{A}) - \Pr[\mathbf{Z} = 1] \cdot H(\mathbf{A} \mid \mathbf{B}, \mathbf{Z} = 1) \\ &\qquad - \Pr[\mathbf{Z} = 0] \cdot H(\mathbf{A} \mid \mathbf{B}, \mathbf{Z} = 0) - H(\mathbf{Z}) \\ &\geq d - (1-\delta)H(1/4)d - \delta d - H(\delta) \\ &= (1-\delta)(1 - H(1/4))d - H(\delta). \end{aligned}$$

2. $I(\mathbf{A} : \mathbf{B}) \leq T \cdot I(\mathbf{A} : \mathbf{B}_1)$.

   Proof. This inequality is essentially due to Jain and Zhang [35, Lemma 5], we include the proof for completeness.

$$\begin{aligned} I(\mathbf{A} : \mathbf{B}) = H(\mathbf{B}) - H(\mathbf{B} \mid \mathbf{A}) &= H(\mathbf{B}) - \sum_{i=1}^{T} H(\mathbf{B}_i \mid \mathbf{A}) \\ &\leq \sum_{i=1}^{T} H(\mathbf{B}_i) - \sum_{i=1}^{T} H(\mathbf{B}_i \mid \mathbf{A}) = \sum_{i=1}^{T} I(\mathbf{A} : \mathbf{B}_i), \end{aligned}$$

   where the second equality used independence of the $\mathbf{B}_i$'s conditioned on $\mathbf{A}$, and the inequality uses Fact 9. Since $I(\mathbf{A} : \mathbf{B}_i) = I(\mathbf{A} : \mathbf{B}_1)$ for all i, we get the inequality.

3. $I(\mathbf{A} : \mathbf{B}_1) = 4\varepsilon$.

   Proof. View $\mathbf{B}_1 = (\mathbf{I}, \mathbf{L})$ as consisting of an index $\mathbf{I} \in \{0, 1, \ldots, d\}$ and a corresponding label $\mathbf{L} \in \{0, 1\}$. With probability $1 - 4\varepsilon$, $(\mathbf{I}, \mathbf{L}) = (0, 0)$. For each $i \in [d]$, with probability $4\varepsilon/d$, $(\mathbf{I}, \mathbf{L}) = (i, \mathbf{A}_i)$. Note that $I(\mathbf{A} : \mathbf{I}) = 0$ because $\mathbf{I}$ is independent of $\mathbf{A}$; $I(\mathbf{A} : \mathbf{L} \mid \mathbf{I} = 0) = 0$; and $I(\mathbf{A} : \mathbf{L} \mid \mathbf{I} = i) = I(\mathbf{A}_i : \mathbf{L} \mid \mathbf{I} = i) = H(\mathbf{A}_i \mid \mathbf{I} = i) - H(\mathbf{A}_i \mid \mathbf{L}, \mathbf{I} = i) = 1 - 0 = 1$ for all $i \in [d]$. We have

$$I(\mathbf{A} : \mathbf{B}_1) = I(\mathbf{A} : \mathbf{I}) + I(\mathbf{A} : \mathbf{L} \mid \mathbf{I}) = \sum_{i=1}^{d} \Pr[\mathbf{I} = i] \cdot I(\mathbf{A} : \mathbf{L} \mid \mathbf{I} = i) = 4\varepsilon.$$

Combining these three steps implies $T = \Omega(d/\varepsilon)$. ◀

3.2.2 Optimal lower bound for classical agnostic learning

▶ **Theorem 13.** *Let $\mathcal{C}$ be a concept class with VC-dim$(\mathcal{C}) = d$. Then for every $\delta \in (0, 1/2)$ and $\varepsilon \in (0, 1/4)$, every (ε, δ)-agnostic learner for $\mathcal{C}$ has sample complexity $\Omega\Big(\frac{d}{\varepsilon^2} + \frac{\log(1/\delta)}{\varepsilon^2}\Big)$.*

Proof. The d-independent part of the lower bound, $T = \Omega(\log(1/\delta)/\varepsilon^2)$, even holds for quantum examples and was proven in Lemma 11. For the other part, the proof is similar to Theorem 12, as follows. Assume an (ε, δ)-agnostic learner for $\mathcal{C}$ that uses T examples. We need to prove $T = \Omega(d/\varepsilon^2)$. For shattered set $\mathcal{S} = \{s_1, \ldots, s_d\} \subseteq \{0, 1\}^n$ and $a \in \{0, 1\}^d$, define distribution D_a on $[d] \times \{0, 1\}$ by $D_a(i, \ell) = (1 + (-1)^{a_i + \ell} 4\varepsilon)/2d$.

Again let random variable $\mathbf{A} \in \{0, 1\}^d$ be uniformly random, corresponding to the values of concept c_a on $\mathcal{S}$, and $\mathbf{B} = \mathbf{B}_1 \ldots \mathbf{B}_T$ be T i.i.d. samples from D_a. Note that c_a is the minimal-error concept from $\mathcal{C}$ w.r.t. D_a, and concept $c_{\tilde{a}}$ has additional error $d_H(a, \tilde{a}) \cdot 4\varepsilon/d$. Accordingly, an (ε, δ)-agnostic learner has to produce (from $\mathbf{B}$) an $h(\mathbf{B}) \in \{0, 1\}^d$, which, with probability at least $1 - \delta$, is $d/4$-close to $\mathbf{A}$. Our three-step analysis is very similar to Theorem 12; only the third step changes:

1. $I(\mathbf{A} : \mathbf{B}) \geq (1 - \delta)(1 - H(1/4))d - H(\delta) = \Omega(d)$.
2. $I(\mathbf{A} : \mathbf{B}) \leq T \cdot I(\mathbf{A} : \mathbf{B}_1)$.
3. $I(\mathbf{A} : \mathbf{B}_1) = 1 - H(1/2 + 2\varepsilon) = O(\varepsilon^2)$.

   Proof. View the D_a-distributed random variable $\mathbf{B}_1 = (\mathbf{I}, \mathbf{L})$ as index $\mathbf{I} \in [d]$ and label $\mathbf{L} \in \{0, 1\}$. The marginal distribution of $\mathbf{I}$ is uniform; conditioned on $\mathbf{I} = i$, the bit $\mathbf{L}$ equals $\mathbf{A}_i$ with probability $1/2 + 2\varepsilon$. Hence

$$I(\mathbf{A} : \mathbf{L} \mid \mathbf{I} = i) = I(\mathbf{A}_i : \mathbf{L} \mid \mathbf{I} = i) = H(\mathbf{A}_i \mid \mathbf{I} = i) - H(\mathbf{A}_i \mid \mathbf{L}, \mathbf{I} = i) = 1 - H(1/2 + 2\varepsilon).$$

   Using Fact 6, we have

$$\begin{aligned} I(\mathbf{A} : \mathbf{B}_1) = I(\mathbf{A} : \mathbf{I}) + I(\mathbf{A} : \mathbf{L} \mid \mathbf{I}) &= \sum_{i=1}^{d} \Pr[\mathbf{I} = i] \cdot I(\mathbf{A} : \mathbf{L} \mid \mathbf{I} = i) \\ &= 1 - H(1/2 + 2\varepsilon) = O(\varepsilon^2). \end{aligned}$$

Combining these three steps implies $T = \Omega(d/\varepsilon^2)$. ◀

In the theorem below, we optimize the constant in the lower bound of the sample complexity in Theorem 13. In learning theory such lower bounds are often stated slightly differently. In order to compare the lower bounds, we introduce the following. We first define an *ε-average agnostic learner* for a concept class $\mathcal{C}$ as a learner that, given access to

T samples from an AEX(D) oracle (for some unknown distribution D), needs to output a hypothesis $h_{\mathbf{XY}}$ (where $(\mathbf{X},\mathbf{Y}) \sim D^T$) that satisfies

$$\mathop{\mathbb{E}}_{(\mathbf{X},\mathbf{Y})\sim D^T}[\mathrm{err}_D(h_{\mathbf{XY}})] - \mathrm{opt}_D(\mathcal{C}) \leq \varepsilon.$$

Lower bounds on the quantity $(\mathbb{E}_{(\mathbf{X},\mathbf{Y})\sim D^T}[\mathrm{err}_D(h_{\mathbf{XY}})] - \mathrm{opt}_D(\mathcal{C}))$ are generally referred to as *minimax lower bounds* in learning theory. For concept class $\mathcal{C}$, Audibert [12, 13] showed that there exists a distribution D, such that if the agnostic learner uses T samples from AEX(D), then

$$\mathop{\mathbb{E}}_{(\mathbf{X},\mathbf{Y})\sim D^T}[\mathrm{err}_D(h_{\mathbf{XY}})] - \mathrm{opt}_D(\mathcal{C}) \geq \frac{1}{6}\sqrt{\frac{d}{T}}.$$

Equivalently, this is a lower bound of $T \geq \frac{d}{36\varepsilon^2}$ on the sample complexity of an ε-average agnostic learner. We obtain a slightly weaker lower bound that is essentially $T \geq \frac{d}{62\varepsilon^2}$:

▶ **Theorem 14.** *Let $\mathcal{C}$ be a concept class with VC-dim($\mathcal{C}$) $= d$. Then for every $\varepsilon \in (0, 1/10]$, there exists a distribution for which every ε-average agnostic learner has sample complexity at least $\frac{d}{\varepsilon^2} \cdot \left(\frac{1}{62} - \frac{\log(2d+2)}{4d}\right)$.*

Proof. The proof is similar to Theorem 13. Assume an ε-average agnostic learner for $\mathcal{C}$ that uses T samples. For shattered set $\mathcal{S} = \{s_1, \ldots, s_d\} \subseteq \{0,1\}^n$ and $a \in \{0,1\}^d$, define distribution D_a on $[d] \times \{0,1\}$ by $D_a(i,\ell) = (1 + (-1)^{a_i+\ell}\beta\varepsilon)/2d$, for some constant $\beta \geq 2$ which we shall pick later.

Again let random variable $\mathbf{A} \in \{0,1\}^d$ be uniformly random, corresponding to the values of concept c_a on $\mathcal{S}$, and $\mathbf{B} = \mathbf{B}_1 \ldots \mathbf{B}_T$ be T i.i.d. samples from D_a. Note that c_a is the minimal-error concept from $\mathcal{C}$ w.r.t. D_a, and concept $c_{\tilde{a}}$ has additional error $d_H(a,\tilde{a}) \cdot \beta\varepsilon/d$. Accordingly, an ε-average agnostic learner has to produce (from $\mathbf{B}$) an $h(\mathbf{B}) \in \{0,1\}^d$, which satisfies $\mathbb{E}_{\mathbf{A},\mathbf{B}}[d_H(\mathbf{A}, h(\mathbf{B}))] \leq d/\beta$.

Our three-step analysis is very similar to Theorem 13; only the first step changes:

1. $I(\mathbf{A} : \mathbf{B}) \geq d(1 - H(1/\beta)) - \log(d+1)$.

   Proof. Define random variable $\mathbf{Z} = d_H(\mathbf{A}, h(\mathbf{B}))$, then $\mathbb{E}[\mathbf{Z}] \leq d/\beta$. Note that given a string $h(\mathbf{B})$ that is ℓ-close to $\mathbf{A}$, $\mathbf{A}$ ranges over a set of only $\binom{d}{\ell} \leq 2^{H(\ell/d)d}$ possible d-bit strings (using Fact 7), hence $H(\mathbf{A} \mid \mathbf{B}, \mathbf{Z} = \ell) \leq H(\mathbf{A} \mid h(\mathbf{B}), \mathbf{Z} = \ell) \leq H(\ell/d)d$. We now lower bound $I(\mathbf{A} : \mathbf{B})$

$$\begin{aligned}
I(\mathbf{A} : \mathbf{B}) &= H(\mathbf{A}) - H(\mathbf{A} \mid \mathbf{B}) \\
&\geq H(\mathbf{A}) - H(\mathbf{A} \mid \mathbf{B}, \mathbf{Z}) - H(\mathbf{Z}) \\
&= d - \sum_{\ell=0}^{d+1} \Pr[\mathbf{Z} = \ell] \cdot H(\mathbf{A} \mid \mathbf{B}, \mathbf{Z} = \ell) - H(\mathbf{Z}) \\
&\geq d - \mathop{\mathbb{E}}_{\ell \in \{0,\ldots,d\}}[H(\ell/d)d] - \log(d+1) && \text{(since } \mathbf{Z} \in \{0,\ldots,d\}\text{)} \\
&\geq d - dH\Big(\frac{\mathbb{E}_\ell[\ell]}{d}\Big) - \log(d+1) && \text{(using Jensen's inequality)} \\
&\geq d - dH(1/\beta) - \log(d+1), && \text{(using } \mathbb{E}[\mathbf{Z}] \leq d/\beta\text{)}
\end{aligned}$$

   where for the third inequality we used the concavity of the binary entropy function to conclude $\mathbb{E}_\ell[H(\ell/d)] \leq H(\mathbb{E}_\ell[\ell]/d)$, and for the fourth inequality we used that $\beta \geq 2$.
2. $I(\mathbf{A} : \mathbf{B}) \leq T \cdot I(\mathbf{A} : \mathbf{B}_1)$.
3. $I(\mathbf{A} : \mathbf{B}_1) = 1 - H(1/2 + \beta\varepsilon/2) \overset{\text{Fact 6}}{\leq} \beta^2\varepsilon^2/\ln 4 + O(\varepsilon^4)$.

Combining these three steps implies

$$T \geq \frac{d \ln 4}{\varepsilon^2} \cdot \Big(\frac{1 - H(1/\beta)}{\beta^2 + O(\varepsilon^2)} - \frac{\log(d+1)}{\beta^2 d + O(d\varepsilon^2)} \Big).$$

Using $\varepsilon \leq 1/10$, $\beta = 4$ to optimize this lower bound, we obtain $T \geq \frac{d}{\varepsilon^2} \cdot \Big(\frac{1}{62} - \frac{\log(2d+2)}{4d} \Big)$. ◀

3.3 Information-theoretic lower bounds on sample complexity: quantum case

Here we will "quantize" the above two classical information-theoretic proofs, yielding lower bounds for quantum sample complexity (in both the PAC and the agnostic setting) that are tight up to a logarithmic factor.

3.3.1 Near-optimal lower bound for quantum PAC learning

▶ **Theorem 15.** *Let $\mathcal{C}$ be a concept class with VC-dim$(\mathcal{C}) = d+1$. Then, for every $\delta \in (0, 1/2)$ and $\varepsilon \in (0, 1/4)$, every (ε, δ)-PAC quantum learner for $\mathcal{C}$ has sample complexity $\Omega\Big(\frac{d}{\varepsilon \log(d/\varepsilon)} + \frac{\log(1/\delta)}{\varepsilon} \Big)$.*

Proof. The proof is analogous to Theorem 12. We use the same distribution D, with the $\mathbf{B}_i$ now being quantum samples: $|\psi_a\rangle = \sum_{i \in \{0,1,\ldots,d\}} \sqrt{D(s_i)} |i, c_a(s_i)\rangle$. The $\mathbf{AB}$-system is now in the following classical-quantum state:

$$\frac{1}{2^d} \sum_{a \in \{0,1\}^d} |a\rangle\langle a| \otimes |\psi_a\rangle\langle\psi_a|^{\otimes T}.$$

The first two steps of our argument are identical to Theorem 12. We only need to re-analyze step 3:

1. $I(\mathbf{A} : \mathbf{B}) \geq (1-\delta)(1 - H(1/4))d - H(\delta) = \Omega(d)$.
2. $I(\mathbf{A} : \mathbf{B}) \leq T \cdot I(\mathbf{A} : \mathbf{B}_1)$.
3. $I(\mathbf{A} : \mathbf{B}_1) \leq H(4\varepsilon) + 4\varepsilon \log(2d) = O(\varepsilon \log(d/\varepsilon))$.

   Proof. Since $\mathbf{AB}$ is a classical-quantum state, we have

   $$I(\mathbf{A} : \mathbf{B}_1) = S(\mathbf{A}) + S(\mathbf{B}_1) - S(\mathbf{A}\mathbf{B}_1) = S(\mathbf{B}_1),$$

   where the first equality follows from definition and the second equality uses $S(\mathbf{A}) = d$ since $\mathbf{A}$ is uniformly distributed in $\{0,1\}^d$, and $S(\mathbf{AB}_1) = d$ since the matrix $\sigma = \frac{1}{2^d} \sum_{a \in \{0,1\}^d} |a\rangle\langle a| \otimes |\psi_a\rangle\langle\psi_a|$ is block diagonal with 2^d rank-1 blocks on the diagonal. It thus suffices to bound the entropy of the singular values of the reduced state of $\mathbf{B}_1$, which is

   $$\rho = \frac{1}{2^d} \sum_{a \in \{0,1\}^d} |\psi_a\rangle\langle\psi_a|.$$

   Let $\sigma_0 \geq \sigma_1 \geq \cdots \geq \sigma_{2d} \geq 0$ be its singular values. Since ρ is a density matrix, these form a probability distribution. Note that the upper-left entry of the matrix $|\psi_a\rangle\langle\psi_a|$ is $D(s_0) = 1 - 4\varepsilon$, hence so is the upper-left entry of ρ. This implies $\sigma_0 \geq 1 - 4\varepsilon$. Consider sampling a number $\mathbf{N} \in \{0, 1, \ldots, 2d\}$ according to the σ-distribution. Let $\mathbf{Z}$ be the indicator random variable for the event $\mathbf{N} \neq 0$, which has probability $1 - \sigma_0 \leq 4\varepsilon$. Note

that $H(\mathbf{N} \mid \mathbf{Z}=0)=0$, because $\mathbf{Z}=0$ implies $\mathbf{N}=0$. Also, $H(\mathbf{N} \mid \mathbf{Z}=1) \leq \log(2d)$, because if $\mathbf{Z}=1$ then $\mathbf{N}$ ranges over $2d$ elements. We now have

$$\begin{aligned} S(\rho) = H(\mathbf{N}) = H(\mathbf{N},\mathbf{Z}) &= H(\mathbf{Z}) + H(\mathbf{N} \mid \mathbf{Z}) \\ &= H(\mathbf{Z}) + \Pr[\mathbf{Z}=0] \cdot H(\mathbf{N} \mid \mathbf{Z}=0) + \Pr[\mathbf{Z}=1] \cdot H(\mathbf{N} \mid \mathbf{Z}=1) \\ &\leq H(4\varepsilon) + 4\varepsilon \log(2d) \\ &= O(\varepsilon \log(d/\varepsilon)). \qquad \text{(using Fact 6)} \end{aligned}$$

Combining these three steps implies $T = \Omega\Big(\frac{d}{\varepsilon \log(d/\varepsilon)}\Big)$. ◀

3.3.2 Near-optimal lower bound for quantum agnostic learning

▶ **Theorem 16.** *Let $\mathcal{C}$ be a concept class with VC-dim$(\mathcal{C}) = d$. Then for every $\delta \in (0,1/2)$ and $\varepsilon \in (0,1/4)$, every (ε,δ)-agnostic quantum learner for $\mathcal{C}$ has sample complexity $\Omega\Big(\frac{d}{\varepsilon^2 \log(d/\varepsilon)} + \frac{\log(1/\delta)}{\varepsilon^2}\Big)$.*

Proof. The proof is analogous to Theorem 13, with the $\mathbf{B}_i$ now being quantum samples for D_a, $|\psi_a\rangle = \sum_{i\in[d],\ell\in\{0,1\}} \sqrt{D_a(i,\ell)}|i,\ell\rangle$. Again we only need to re-analyze step 3:

1. $I(\mathbf{A}:\mathbf{B}) \geq (1-\delta)(1-H(1/4))d - H(\delta) = \Omega(d)$.
2. $I(\mathbf{A}:\mathbf{B}) \leq T \cdot I(\mathbf{A}:\mathbf{B}_1)$.
3. $I(\mathbf{A}:\mathbf{B}_1) = O(\varepsilon^2 \log(d/\varepsilon))$.

Proof (of step 3). As in step 3 of the proof of Theorem 15, it suffices to upper bound the entropy of

$$\rho = \frac{1}{2^d} \sum_{a\in\{0,1\}^d} |\psi_a\rangle\langle\psi_a|.$$

We now lower bound the largest singular value of ρ. Consider $|\psi\rangle = \frac{1}{\sqrt{2d}} \sum_{i\in[d],\ell\in\{0,1\}} |i,\ell\rangle$.

$$\langle\psi|\psi_a\rangle = \frac{1}{d}\sum_{i\in[d]} \frac{1}{2}\Big(\sqrt{1+4\varepsilon} + \sqrt{1-4\varepsilon}\Big) = \frac{1}{2}\Big(\sqrt{1+4\varepsilon} + \sqrt{1-4\varepsilon}\Big) \geq 1 - 2\varepsilon^2 - O(\varepsilon^4),$$

where the last inequality used the Taylor series expansion of $\sqrt{1+x}$. This implies that the largest singular value of ρ is at least

$$\langle\psi|\rho|\psi\rangle = \frac{1}{2^d} \sum_{a\in\{0,1\}^d} |\langle\psi|\psi_a\rangle|^2 \geq 1 - 4\varepsilon^2 - O(\varepsilon^4).$$

We can now finish as in step 3 of the proof of Theorem 15:

$$I(\mathbf{A}:\mathbf{B}_1) \leq S(\rho) \leq H(4\varepsilon^2) + 4\varepsilon^2 \log(2d) \overset{\text{Fact 6}}{=} O(\varepsilon^2 \log(d/\varepsilon)).$$

Combining these three steps implies $T = \Omega\Big(\frac{d}{\varepsilon^2 \log(d/\varepsilon)}\Big)$. ◀

4 A lower bound by analysis of state identification

In this section we present a tight lower bound on quantum sample complexity for both the PAC and the agnostic learning settings, using ideas from Fourier analysis to analyze the performance of the Pretty Good Measurement. The core of both lower bounds is the following theorem.

▶ **Theorem 17.** *For $m \geq 10$, let $f : \{0,1\}^m \to \mathbb{R}$ be defined as $f(z) = (1 - \beta\frac{|z|}{m})^T$ for some $\beta \in (0,1]$ and $T \in [1, m/(e^3\beta)]$. For $k \leq m$, let $M \in \mathbb{F}_2^{m\times k}$ be a matrix with rank k. Suppose $A \in \mathbb{R}^{2^k\times 2^k}$ is defined as $A(x,y) = (f \circ M)(x+y)$ for $x, y \in \{0,1\}^k$, then*

$$\sqrt{A}(x,x) \leq \frac{2\sqrt{e}}{2^{k/2}}\Big(1 - \frac{\beta}{2}\Big)^{T/2} e^{11T^2\beta^2/m + \sqrt{Tm\beta}} \qquad \text{for all } x \in \{0,1\}^k.$$

Proof. The structure of the proof is to first diagonalize A, relating its eigenvalues to the Fourier coefficients of f. This allows to calculate the diagonal entries of $\sqrt{A}$ exactly in terms of those Fourier coefficients. We then upper bound those Fourier coefficients using a combinatorial argument.

We first observe the well-known relation between the eigenvalues of a matrix P defined as $P(x,y) = g(x+y)$ for $x, y \in \{0,1\}^k$, and the Fourier coefficients of g.

▶ **Claim 18.** *Suppose $g : \{0,1\}^k \to \mathbb{R}$ and $P \in \mathbb{R}^{2^k\times 2^k}$ is defined as $P(x,y) = g(x+y)$, then the eigenvalues of P are $\{2^k\widehat{g}(Q) : Q \in \{0,1\}^k\}$.*

Proof. Let $H \in \mathbb{R}^{2^k\times 2^k}$ be the matrix defined as $H(x,y) = (-1)^{x\cdot y}$ for $x, y \in \{0,1\}^k$. It is easy to see that $H^{-1}(x,y) = (-1)^{x\cdot y}/2^k$. We now show that H diagonalizes P:

$$\begin{aligned}
(HPH^{-1})(x,y) &= \frac{1}{2^k}\sum_{z_1,z_2\in\{0,1\}^k} (-1)^{z_1\cdot x + z_2\cdot y} g(z_1+z_2) \\
&= \frac{1}{2^k}\sum_{z_1,z_2,Q\in\{0,1\}^k} (-1)^{z_1\cdot x + z_2\cdot y}\widehat{g}(Q)(-1)^{Q\cdot(z_1+z_2)} \\
&\qquad\qquad \text{(Fourier expansion of } g\text{)} \\
&= \frac{1}{2^k}\sum_{Q\in\{0,1\}^k}\widehat{g}(Q)\sum_{z_1\in\{0,1\}^k}(-1)^{(x+Q)\cdot z_1}\sum_{z_2\in\{0,1\}^k}(-1)^{(y+Q)\cdot z_2} \\
&= 2^k\widehat{g}(x)\delta_{x,y} \qquad\qquad \text{(using } \textstyle\sum_{z\in\{0,1\}^k}[(-1)^{(a+b)\cdot z}] = 2^k\delta_{a,b}\text{)}
\end{aligned}$$

The eigenvalues of P are the diagonal entries, $\{2^k\widehat{g}(Q) : Q \in \{0,1\}^k\}$. ◀

We now relate the diagonal entries of $\sqrt{A}$ to the Fourier coefficients of f:

▶ **Claim 19.** *For all $x \in \{0,1\}^k$, we have*

$$\sqrt{A}(x,x) = \frac{1}{2^{k/2}}\sum_{Q\in\{0,1\}^k}\sqrt{\sum_{S\in\{0,1\}^m : M^tS = Q}\widehat{f}(S)}\,.$$

Proof. Since $A(x,y) = (f\circ M)(x+y)$, by Claim 18 it follows that H (as defined in the proof of Claim 18) diagonalizes A and the eigenvalues of A are $\{2^k\widehat{f\circ M}(Q) : Q \in \{0,1\}^k\}$. Hence, we have

$$\sqrt{A} = H^{-1}\cdot \mathrm{diag}\Big(\Big\{\sqrt{2^k\widehat{f\circ M}(Q)} : Q \in \{0,1\}^k\Big\}\Big)\cdot H,$$

and the diagonal entries of $\sqrt{A}$ are

$$\sqrt{A}(x,x) = \frac{1}{2^{k/2}}\sum_{Q\in\{0,1\}^k}\sqrt{\widehat{f\circ M}(Q)} \overset{\text{Claim 4}}{=} \frac{1}{2^{k/2}}\sum_{Q\in\{0,1\}^k}\sqrt{\sum_{S\in\{0,1\}^m : M^tS = Q}\widehat{f}(S)}.$$ ◀

In the following lemma, we give an upper bound on the Fourier coefficients of f, which in turn (from the claim above) gives an upper bound on the diagonal entries of $\sqrt{A}$.

▶ **Lemma 20.** *For $\beta \in (0,1]$, the Fourier coefficients of $f : \{0,1\}^m \to \mathbb{R}$ defined as $f(z) = (1-\beta\frac{|z|}{m})^T$, satisfy*

$$0 \le \widehat{f}(S) \le 4e\Big(1-\frac{\beta}{2}\Big)^T\Big(\frac{T\beta}{m}\Big)^q e^{22T^2\beta^2/m}, \quad \text{for all } S \text{ such that } |S| = q.$$

Proof. In order to see why the Fourier coefficients of f are non-negative, we first define the set $U = \{u_x^{\otimes T}\}_{x\in\{0,1\}^m}$ where $u_x = \sqrt{1-\beta}|0,0\rangle + \sqrt{\beta/m}\sum_{i\in[m]}|i,x_i\rangle$. Let V be the $2^m \times 2^m$ Gram matrix for the set U. For $x,y \in \{0,1\}^m$, we have

$$\begin{aligned} V(x,y) = (u_x^* u_y)^T &= \Big(1-\beta+\frac{\beta}{m}\sum_{i=1}^m \langle x_i|y_i\rangle\Big)^T \\ &= \Big(1-\beta+\frac{\beta}{m}(m-|x+y|)\Big)^T = \Big(1-\beta\frac{|x+y|}{m}\Big)^T = f(x+y). \end{aligned}$$

By Claim 18, the eigenvalues of the Gram matrix V are $\{2^m\widehat{f}(S) : S \in \{0,1\}^m\}$. Since the Gram matrix is psd, its eigenvalues are non-negative, which implies that $\widehat{f}(S) \ge 0$ for all $S \in \{0,1\}^m$.

We now prove the upper bound in the lemma. By definition,

$$\begin{aligned} \widehat{f}(S) &= \mathop{\mathbb{E}}_{z\in\{0,1\}^m}\Big[\Big(1-\beta\frac{|z|}{m}\Big)^T(-1)^{S\cdot z}\Big] \\ &= \mathop{\mathbb{E}}_{z\in\{0,1\}^m}\Big[\Big(1-\frac{\beta}{2}+\frac{\beta}{2m}\sum_{i=1}^m(-1)^{z_i}\Big)^T(-1)^{S\cdot z}\Big] \qquad (\text{since } |z| = \textstyle\sum_{i\in[m]}\frac{1-(-1)^{z_i}}{2}) \\ &= \sum_{\ell=0}^T\binom{T}{\ell}\Big(1-\frac{\beta}{2}\Big)^{T-\ell}\Big(\frac{\beta}{2m}\Big)^\ell \mathop{\mathbb{E}}_{z\in\{0,1\}^m}\Big[\sum_{i_1,\dots,i_\ell=1}^m (-1)^{z\cdot(e_{i_1}+\cdots+e_{i_\ell}+S)}\Big] \\ &= \sum_{\ell=0}^T\binom{T}{\ell}\Big(1-\frac{\beta}{2}\Big)^{T-\ell}\Big(\frac{\beta}{2m}\Big)^\ell \sum_{i_1,\dots,i_\ell=1}^m 1_{[e_{i_1}+\cdots+e_{i_\ell}=S]} \\ &\qquad\qquad (\text{using } \mathbb{E}_{z\in\{0,1\}^m}[(-1)^{(z_1+z_2)\cdot z}] = \delta_{z_1,z_2}) \end{aligned}$$

We will use the following claim to upper bound the combinatorial sum in the quantity above.

▶ **Claim 21.** *Fix $S \in \{0,1\}^m$ with Hamming weight $|S| = q$. For every $\ell \in \{q,\dots,T\}$, we have*

$$\sum_{i_1,\dots,i_\ell=1}^m 1_{[e_{i_1}+\cdots+e_{i_\ell}=S]} \le \begin{cases} \ell!\cdot m^{(\ell-q)/2}\Big/\Big(2^{(\ell-q)/2}((\ell-q)/2)!\Big) & \text{if } (\ell-q) \text{ is even} \\ 0 & \text{otherwise} \end{cases}$$

Proof. Since $|S| = q$, we can write $S = e_{r_1}+\cdots+e_{r_q}$ for distinct $r_1,\dots,r_q \in [m]$. There are $\binom{\ell}{q}$ ways to pick q indices in $(i_1,\dots,i_\ell)$ (w.l.o.g. let them be $i_1,\dots,i_q$) and there are $q!$ factorial ways to assign $(r_1,\dots,r_q)$ to $(i_1,\dots,i_q)$. It remains to count the number of ways that we can assign values to the remaining indices $i_{q+1},\dots,i_\ell$ such that $e_{i_{q+1}}+\cdots+e_{i_\ell} = 0$. If $\ell-q$ is odd then this number is 0, so from now on assume $\ell-q$ is even. We upper bound the number of such assignments by partitioning the $\ell-q$ indices into pairs and assigning the same value to both indices in each pair.

We first count the number of ways to partition a set of $\ell-q$ indices into subsets of size 2. This number is exactly $(\ell-q)!\Big(2^{(\ell-q)/2}((\ell-q)/2)!\Big)^{-1}$. Furthermore, there are m possible values that can be assigned to the pair of indices in each of the $(\ell-q)/2$ subsets such that

$e_i + e_j = 0$ within each subset. Note that assigning m possible values to each pair of indices in the $(\ell - q)/2$ subsets overcounts, but this rough upper bound is sufficient for our purposes.

Combining the three arguments, we conclude

$$\sum_{i_1,\ldots,i_\ell=1}^{d} 1_{[e_{i_1}+\cdots+e_{i_\ell}=S]} \leq \binom{\ell}{q} q! \cdot (\ell-q)! \cdot m^{(\ell-q)/2} \Big/ \Big(2^{(\ell-q)/2}((\ell-q)/2)!\Big).$$

which yields the claim. ◀

Continuing with the evaluation of the Fourier coefficient and using the claim above, we have

$$\begin{aligned}
\widehat{f}(S) &= \sum_{\ell=0}^{T} \binom{T}{\ell} \Big(1-\frac{\beta}{2}\Big)^{T-\ell} \Big(\frac{\beta}{2m}\Big)^{\ell} \sum_{i_1,\ldots,i_\ell=1}^{m} 1_{[e_{i_1}+\cdots+e_{i_\ell}=S]} \\
&\leq \sum_{\ell=q}^{T} \binom{T}{\ell} \Big(1-\frac{\beta}{2}\Big)^{T-\ell} \Big(\frac{\beta}{2m}\Big)^{\ell} \ell! \cdot m^{(\ell-q)/2} \Big/ \Big(2^{(\ell-q)/2}\Big(\frac{\ell-q}{2}\Big)!\Big) \qquad \text{(by Claim 21)} \\
&= \Big(1-\frac{\beta}{2}\Big)^{T} \Big(\frac{2}{m}\Big)^{q/2} \sum_{\ell=q}^{T} \binom{T}{\ell} \ell! \Big(\frac{\beta}{m(2-\beta)}\Big)^{\ell} \Big(\frac{m}{2}\Big)^{\ell/2} \Big/ \Big(\frac{\ell-q}{2}\Big)! \\
&\leq \Big(1-\frac{\beta}{2}\Big)^{T} \Big(\frac{2}{m}\Big)^{q/2} \sum_{\ell=q}^{T} \Big(T \cdot \frac{\beta}{m} \cdot \sqrt{\frac{m}{2}}\Big)^{\ell} \Big/ \Big(\frac{\ell-q}{2}\Big)! \quad \text{(since } \beta < 1 \text{ and } \tbinom{T}{\ell}\ell! \leq T^\ell\text{)} \\
&= \Big(1-\frac{\beta}{2}\Big)^{T} \Big(\frac{T\beta}{m}\Big)^{q} \sum_{r=0}^{T-q} \Big(\frac{T\beta}{\sqrt{2m}}\Big)^{r} \frac{1}{(r/2)!} \qquad \text{(substituting } r \leftarrow (\ell-q)\text{)} \\
&\leq \Big(1-\frac{\beta}{2}\Big)^{T} \Big(\frac{T\beta}{m}\Big)^{q} \sum_{r=0}^{T-q} \Big(\frac{T\beta}{\sqrt{2m}}\Big)^{r} \frac{e^{r/2}}{(r/2)^{r/2}} \qquad \text{(using } n! \geq (n/e)^n\text{)} \\
&= \Big(1-\frac{\beta}{2}\Big)^{T} \Big(\frac{T\beta}{m}\Big)^{q} \sum_{r=0}^{T-q} \Big(\frac{\sqrt{e}T\beta}{\sqrt{mr}}\Big)^{r} \\
&\leq \Big(1-\frac{\beta}{2}\Big)^{T} \Big(\frac{T\beta}{m}\Big)^{q} \sum_{r=0}^{T} \Big(\frac{\sqrt{e}T\beta}{\sqrt{mr}}\Big)^{r} \qquad \text{(since the summands are } \geq 0\text{)} \\
&= \Big(1-\frac{\beta}{2}\Big)^{T} \Big(\frac{T\beta}{m}\Big)^{q} \Big(\sum_{r=0}^{\lceil e^3T^2\beta^2/m\rceil} \Big(\frac{\sqrt{e}T\beta}{\sqrt{mr}}\Big)^{r} + \sum_{r=\lceil e^3T^2\beta^2/m\rceil+1}^{T} \Big(\frac{\sqrt{e}T\beta}{\sqrt{mr}}\Big)^{r} \Big).
\end{aligned}$$

Note that by the assumptions of the theorem, $T^2e^3\beta^2/m \leq T\beta \leq T$, which allowed us to split the sum into two pieces in the last equality. At this point, we upper bound both pieces in the last equation separately. For the first piece, using Claim 5 it follows that $\left(\frac{\sqrt{e}T\beta}{\sqrt{mr}}\right)^r$ is maximized at $r = \lceil T^2\beta^2/m \rceil$. Hence we get

$$\sum_{r=0}^{\lceil e^3T^2\beta^2/m\rceil} \Big(\frac{\sqrt{e}T\beta}{\sqrt{mr}}\Big)^{r} \leq \Big(2 + \frac{e^3T^2\beta^2}{m}\Big) e^{\lceil T^2\beta^2/m\rceil/2} \leq 2e^{22T^2\beta^2/m+1}, \tag{6}$$

where the first inequality uses Claim 5 and the second inequality uses $2 + x \leq 2e^x$ for $x \geq 0$ and $e^3 + 1/2 \leq 22$. For the second piece, we use

$$\sum_{r=\lceil e^3T^2\beta^2/m\rceil+1}^{T} \Big(\frac{\sqrt{e}T\beta}{\sqrt{mr}}\Big)^{r} \leq \sum_{r=\lceil e^3T^2\beta^2/m\rceil+1}^{T} \Big(\frac{1}{e}\Big)^{r} \leq \sum_{r=1}^{T} \Big(\frac{1}{e}\Big)^{r} = \frac{1-e^{-T}}{e-1} \leq 2/3. \tag{7}$$

So we finally get

$$\begin{aligned}\widehat{f}(S) &\le \Big(1-\frac{\beta}{2}\Big)^T\Big(\frac{T\beta}{m}\Big)^q\Big(2e^{22T^2\beta^2/m+1}+2/3\Big) && \text{(using Eq. (6), (7))}\\ &\le 4e\Big(1-\frac{\beta}{2}\Big)^T\Big(\frac{T\beta}{m}\Big)^q e^{22T^2\beta^2/m} && \text{(since } 22T^2\beta^2/m>0\text{)}\end{aligned}$$

◀

The theorem follows by putting together Claim 19 and Lemma 20:

$$\begin{aligned}\sqrt{A}(x,x) &= \frac{1}{2^{k/2}}\sum_{Q\in\{0,1\}^k}\sqrt{\sum_{S\in\{0,1\}^m:M^tS=Q}\widehat{f}(S)} && \text{(using Claim 19)}\\ &\le \frac{1}{2^{k/2}}\sum_{Q\in\{0,1\}^k}\sum_{S\in\{0,1\}^m:M^tS=Q}\sqrt{\widehat{f}(S)} && \text{(using lower bound from Lemma 20)}\\ &= \frac{1}{2^{k/2}}\sum_{S\in\{0,1\}^m}\sqrt{\widehat{f}(S)} && (\cup_Q\{S:M^tS=Q\}=\{0,1\}^m \text{ since rank}(M)=k)\\ &= \frac{1}{2^{k/2}}\sum_{q=0}^{m}\sum_{S\in\{0,1\}^m:|S|=q}\sqrt{\widehat{f}(S)}\\ &\le \frac{2\sqrt{e}}{2^{k/2}}\Big(1-\frac{\beta}{2}\Big)^{T/2}e^{11T^2\beta^2/m}\sum_{q=0}^{m}\binom{m}{q}\Big(\frac{T\beta}{m}\Big)^{q/2} && \text{(using Lemma 20)}\\ &= \frac{2\sqrt{e}}{2^{k/2}}\Big(1-\frac{\beta}{2}\Big)^{T/2}e^{11T^2\beta^2/m}\Big(1+\sqrt{\frac{T\beta}{m}}\Big)^m && \text{(using binomial theorem)}\\ &\le \frac{2\sqrt{e}}{2^{k/2}}\Big(1-\frac{\beta}{2}\Big)^{T/2}e^{11T^2\beta^2/m+\sqrt{Tm\beta}}. && \text{(using } (1+x)^t\le e^{xt} \text{ for } x,t\ge 0\text{)}\end{aligned}$$

◀

4.1 Optimal lower bound for quantum PAC learning

We can now prove our tight lower bound on quantum sample complexity in the PAC model:

▶ **Theorem 22.** *Let $\mathcal{C}$ be a concept class with VC-dim$(\mathcal{C})=d+1$, for sufficiently large d. Then for every $\delta\in(0,1/2)$ and $\varepsilon\in(0,1/20)$, every (ε,δ)-PAC quantum learner for $\mathcal{C}$ has sample complexity $\Omega\Big(\frac{d}{\varepsilon}+\frac{1}{\varepsilon}\log\frac{1}{\delta}\Big)$.*

Proof. The d-independent part of the lower bound is Lemma 10. To prove the d-dependent part, define a distribution D on a set $\mathcal{S}=\{s_0,\ldots,s_d\}\subseteq\{0,1\}^n$ that is shattered by $\mathcal{C}$ as follows: $D(s_0)=1-20\varepsilon$ and $D(s_i)=20\varepsilon/d$ for all $i\in[d]$.

Now consider a $[d,k,r]_2$ linear code (for $k\ge d/4$, distance $r\ge d/8$) as shown to exist in Theorem 3 with the generator matrix $M\in\mathbb{F}_2^{d\times k}$ of rank k. Let $\{Mx:x\in\{0,1\}^k\}\subseteq\{0,1\}^d$ be the set of codewords in this linear code; these satisfy $d_H(Mx,My)\ge d/8$ whenever $x\ne y$. For each $x\in\{0,1\}^k$, let c^x be a concept defined on the shattered set as: $c^x(s_0)=0$ and $c^x(s_i)=(Mx)_i$ for all $i\in[d]$. The existence of such concepts in $\mathcal{C}$ follows from the fact that $\mathcal{S}$ is shattered by $\mathcal{C}$. From the distance property of the code, we have $\Pr_{s\sim D}[c^x(s)\ne c^y(s)]\ge\frac{20\varepsilon}{d}\frac{d}{8}=5\varepsilon/2$. This in particular implies that an (ε,δ)-PAC quantum learner that tries to ε-approximate a concept from $\{c^x:x\in\{0,1\}^k\}$ should successfully *identify* that concept with probability at least $1-\delta$.

We now consider the following state identification problem: for $x\in\{0,1\}^k$, denote $|\psi_x\rangle=\sum_{i\in\{0,\ldots,d\}}\sqrt{D(s_i)}|s_i,c^x(s_i)\rangle$. Let the (ε,δ)-PAC quantum sample complexity be T.

Assume $T \leq d/(20e^3\varepsilon)$, since otherwise $T \geq \Omega(d/\varepsilon)$ and the theorem follows. Suppose the learner has knowledge of the ensemble $\mathcal{E} = \{(2^{-k}, |\psi_x\rangle^{\otimes T}) : x \in \{0,1\}^k\}$, and is given $|\psi_x\rangle^{\otimes T} \in \mathcal{E}$ for a uniformly random x. The learner would like to maximize the average probability of success to identify the given state. For this problem, we prove a lower bound on T using the PGM defined in Section 2.6. In particular, we show that using the PGM, if a learner successfully identifies the states in $\mathcal{E}$, then $T = \Omega(d/\varepsilon)$. Since the PGM is the optimal measurement[6] that the learner could have performed, the result follows. The following lemma makes this lower bound rigorous and will conclude the proof of the theorem.

▶ **Lemma 23.** *For every $x \in \{0,1\}^k$, let $|\psi_x\rangle = \sum_{i\in\{0,\ldots,d\}} \sqrt{D(s_i)}|s_i, c^x(s_i)\rangle$, and $\mathcal{E} = \{(2^{-k}, |\psi_x\rangle^{\otimes T}) : x \in \{0,1\}^k\}$. Then*[7]

$$P^{PGM}(\mathcal{E}) \leq \frac{4e}{2^{d/4+T\varepsilon}} e^{8800T^2\varepsilon^2/d + 4\sqrt{5Td\varepsilon}}.$$

Before we prove the lemma, we first show why it implies the theorem. Since we observed above that $P^{opt}(\mathcal{E}) = P^{PGM}(\mathcal{E})$, a good learner satisfies $P^{PGM}(\mathcal{E}) = \Omega(1)$ (say for $\delta = 1/4$), which in turn implies

$$\Omega(\max\{d, T\varepsilon\}) \leq O(\min\{T^2\varepsilon^2/d, \sqrt{Td\varepsilon}\}).$$

Note that if $T\varepsilon$ maximizes the left-hand side, then $d \leq T\varepsilon$ and hence $T \geq \Omega(d/\varepsilon)$. The remaining cases are $\Omega(d) \leq T^2\varepsilon^2/d$ and $\Omega(d) \leq \sqrt{Td\varepsilon}$. Both these statements give us $T \geq \Omega(d/\varepsilon)$. Hence the theorem follows, and it remains to prove Lemma 23:

Proof. Let $\mathcal{E}' = \{2^{-k/2}|\psi_x\rangle^{\otimes T} : x \in \{0,1\}^k\}$ and G be the $2^k \times 2^k$ Gram matrix for $\mathcal{E}'$. As we saw in Section 2.6, the success probability of identifying the states in the ensemble $\mathcal{E}$ using the PGM is

$$P^{PGM}(\mathcal{E}) = \sum_{x\in\{0,1\}^k} \sqrt{G}(x,x)^2.$$

For all $x, y \in \{0,1\}^k$, the entries of the Gram matrix G can be written as:

$$\begin{aligned} G(x,y) = \frac{1}{2^k}\langle\psi_x|\psi_y\rangle^T &= \frac{1}{2^k}\Big((1-20\varepsilon) + \frac{20\varepsilon}{d}\sum_{i=1}^{d}\langle c^x(s_i)|c^y(s_i)\rangle\Big)^T \\ &= \frac{1}{2^k}\Big((1-20\varepsilon) + \frac{20\varepsilon}{d}(d - d_H(Mx, My))\Big)^T \\ &= \frac{1}{2^k}\Big(1 - \frac{20\varepsilon}{d}d_H(Mx, My)\Big)^T, \end{aligned}$$

where Mx, $My \in \{0,1\}^d$ are codewords in the linear code defined earlier. Define $f : \{0,1\}^d \to \mathbb{R}$ as $f(z) = (1 - \frac{20\varepsilon}{d}|z|)^T$, and let $A(x,y) = (f \circ M)(x+y)$ for $x, y \in \{0,1\}^k$. Note that $G = A/2^k$. Since we assumed $T \leq d/(20e^3\varepsilon)$, we can use Theorem 17 (by choosing $m = d$

[6] For $x \in \{0,1\}^k$, define unitary $U_{c^x} : |s_i, b\rangle \to |s_i, b + c^x(s_i)\rangle$ for all $i \in \{0,\ldots,d\}$. The ensemble $\mathcal{E}$ is generated by applying $\{U_{c^x}\}_{x\in\{0,1\}^k}$ to $|\varphi\rangle = \sum_{i\in\{0,\ldots,d\}} \sqrt{D(s_i)}|s_i, 0\rangle$. View $c^x = (0, Mx) \in \{0,1\}^{d+1}$ as a concatenated string where Mx is a codeword of the $[d,k,r]_2$ code. Since the 2^k codewords of the $[d,k,r]_2$ code form a linear subspace, $\{U_{c^x}\}_{x\in\{0,1\}^k}$ is an Abelian group. From the discussion in Section 2.6, we conclude that the PGM is the optimal measurement for this state identification problem.

[7] We made no attempt to optimize the constants here.

and $\beta = 20\varepsilon$) to upper bound the success probability of successfully identifying the states in the ensemble $\mathcal{E}$ using the PGM.

$$\begin{aligned}
P^{PGM}(\mathcal{E}) &= \sum_{x\in\{0,1\}^k} \sqrt{G}(x,x)^2 \\
&= \frac{1}{2^k} \sum_{x\in\{0,1\}^k} \sqrt{A}(x,x)^2 && \text{(since } G = A/2^k\text{)} \\
&\leq \frac{4e}{2^k}\Big(1-\frac{\beta}{2}\Big)^T e^{22T^2\beta^2/d+2\sqrt{Td\beta}} && \text{(using Theorem 17)} \\
&= \frac{4e}{2^k}\Big(1-10\varepsilon\Big)^T e^{8800T^2\varepsilon^2/d+4\sqrt{5Td\varepsilon}} && \text{(substituting } \beta = 20\varepsilon\text{)} \\
&\leq \frac{4e}{2^{k+T\varepsilon}} e^{8800T^2\varepsilon^2/d+4\sqrt{5Td\varepsilon}} && \text{(using } (1-10\varepsilon)^T \leq e^{-10\varepsilon T} \leq 2^{-\varepsilon T}\text{)}
\end{aligned}$$

The lemma follows by observing that $k \geq d/4$. ◀◀

4.2 Optimal lower bound for quantum agnostic learning

We now use the same approach to obtain a tight lower bound on quantum sample complexity in the *agnostic* setting.

▶ **Theorem 24.** *Let $\mathcal{C}$ be a concept class with VC-dim$(\mathcal{C}) = d$, for sufficiently large d. Then for every $\delta \in (0, 1/2)$ and $\varepsilon \in (0, 1/10)$, every (ε,δ)-agnostic quantum learner for $\mathcal{C}$ has sample complexity $\Omega\Big(\frac{d}{\varepsilon^2} + \frac{1}{\varepsilon^2}\log\frac{1}{\delta}\Big)$.*

Proof. The d-independent part of the lower bound is Lemma 11. For the d-dependent term in the lower bound, consider a $[d,k,r]_2$ linear code (for $k \geq d/4$, distance $r \geq d/8$) as shown to exist in Theorem 3, with generator matrix $M \in \mathbb{F}_2^{d\times k}$ of rank k. Let $\{Mx : x \in \{0,1\}^k\} \subseteq \{0,1\}^d$ be the set of 2^k codewords in this linear code; these satisfy $d_H(Mx, My) \geq d/8$ whenever $x \neq y$. To each codeword $x \in \{0,1\}^k$ we associate a distribution D_x as follows:

$$D_x(s_i, b) = \frac{1}{d}\Big(\frac{1}{2} + \frac{1}{2}(-1)^{(Mx)_i+b}\alpha\Big), \qquad \text{for } (i,b) \in [d]\times\{0,1\},$$

where $\mathcal{S} = \{s_1,\ldots,s_d\}$ is a set that is shattered by $\mathcal{C}$, and α is a parameter which we shall pick later. Let $c^x \in \mathcal{C}$ be a concept that labels $\mathcal{S}$ according to $Mx \in \{0,1\}^d$. The existence of such $c^x \in \mathcal{C}$ follows from the fact that $\mathcal{S}$ is shattered by $\mathcal{C}$. Note that c^x is the minimal-error concept in $\mathcal{C}$ w.r.t. D_x. A learner that labels $\mathcal{S}$ according to some string $\ell \in \{0,1\}^d$ has additional error $d_H(Mx,\ell)\cdot\alpha/d$ compared to c^x. This in particular implies that an (ε,δ)-agnostic quantum learner has to find (with probability at least $1-\delta$) an ℓ such that $d_H(Mx,\ell) \leq d\varepsilon/\alpha$. We pick $\alpha = 20\varepsilon$ and we get $d_H(Mx,\ell) \leq d/20$. However, since Mx was a codeword of a $[d,k,r]_2$ code with distance $r \geq d/8$, finding an ℓ satisfying $d_H(Mx,\ell) \leq d/20$ is equivalent to *identifying* Mx, and hence x.

Now consider the following state identification problem: for $x \in \{0,1\}^k$, let $|\psi_x\rangle = \sum_{(i,b)\in[d]\times\{0,1\}} \sqrt{D_x(s_i,b)}|s_i,b\rangle$. Let the (ε,δ)-agnostic quantum sample complexity be T. Assume $T \leq d/(100e^3\varepsilon^2)$, since otherwise $T \geq \Omega(d/\varepsilon^2)$ and the theorem follows. Suppose the learner has knowledge of the ensemble $\mathcal{E} = \{(2^{-k}, |\psi_x\rangle^{\otimes T}) : x \in \{0,1\}^k\}$, and is given $|\psi_x\rangle^{\otimes T} \in \mathcal{E}$ for uniformly random x. The learner would like to maximize the average probability of success to identify the given state. For this problem, we prove a lower bound

on T using the PGM defined in Section 2.6. In particular, we show that using the PGM, if a learner successfully identifies the states in $\mathcal{E}$, then $T = \Omega(d/\varepsilon^2)$. Since the PGM is the optimal measurement[8] that the learner could have performed, the result follows. The following lemma makes this lower bound rigorous and will conclude the proof of the theorem.

▶ **Lemma 25.** *For $x \in \{0,1\}^k$, let $|\psi_x\rangle = \sum_{(i,b)\in[d]\times\{0,1\}} \sqrt{D_x(s_i,b)}|s_i,b\rangle$, and $\mathcal{E} = \{(2^{-k}, |\psi_x\rangle^{\otimes T}) : x \in \{0,1\}^k\}$. Then*

$$P^{PGM}(\mathcal{E}) \leq \frac{4e}{e^{(d\ln 2)/4+25T\varepsilon^2}} e^{220000T^2\varepsilon^4/d+20\sqrt{Td\varepsilon^2}}.$$

Before we prove the lemma, we first show why it implies the theorem. Since we observed above that $P^{opt}(\mathcal{E}) = P^{PGM}(\mathcal{E})$, a good learner satisfies $P^{PGM}(\mathcal{E}) = \Omega(1)$ (say for $\delta = 1/4$), which in turn implies

$$\Omega(\max\{d, T\varepsilon^2\}) \leq O(\min\{T^2\varepsilon^4/d, \sqrt{Td\varepsilon^2}\}).$$

Like in the proof of Theorem 22, this implies a lower bound of $T = \Omega(d/\varepsilon^2)$ and proves the theorem. It remains to prove Lemma 25:

Proof. Let $\mathcal{E}' = \{2^{-k/2}|\psi_x\rangle^{\otimes T} : x \in \{0,1\}^k\}$ and G be the $2^k \times 2^k$ Gram matrix for the set $\mathcal{E}'$. As we saw in Section 2.6, the success probability of identifying the states in the ensemble $\mathcal{E}$ using the PGM is

$$P^{PGM}(\mathcal{E}) = \sum_{x\in\{0,1\}^k} \sqrt{G}(x,x)^2.$$

For all $x, y \in \{0,1\}^k$, the entries of G can be written as:

$$\begin{aligned}
2^k \cdot G(x,y) &= \langle\psi_x|\psi_y\rangle^T \\
&= \Big(\sum_{(i,b)\in[d]\times\{0,1\}} \sqrt{D_x(i,b)D_y(i,b)}\Big)^T \\
&= \Big(\frac{1}{2d}\sum_{(i,b)\in[d]\times\{0,1\}} \sqrt{(1+10\varepsilon(-1)^{(Mx)_i+b})(1+10\varepsilon(-1)^{(My)_i+b})}\Big)^T \\
&= \Big(\frac{1}{2d}\sum_{\substack{(i,b):\\(Mx)_i=(My)_i}} (1+10\varepsilon(-1)^{(Mx)_i+b}) + \frac{1}{2d}\sum_{\substack{(i,b):\\(Mx)_i\neq(My)_i}} \sqrt{1-100\varepsilon^2}\Big)^T \\
&= \Big(\frac{d-d_H(Mx,My)}{d} + \frac{\sqrt{1-100\varepsilon^2}}{d} d_H(Mx,My)\Big)^T \\
&= \Big(1 - \frac{1-\sqrt{1-100\varepsilon^2}}{d} d_H(Mx,My)\Big)^T.
\end{aligned}$$

where we used $\alpha = 20\varepsilon$ in the third equality.

Let $\beta = 1 - \sqrt{1-100\varepsilon^2}$, which is at most 1 for $\varepsilon \leq 1/10$. Define $f : \{0,1\}^d \to \mathbb{R}$ as $f(z) = (1 - \frac{\beta}{d}|z|)^T$, and let $A(x,y) = (f \circ M)(x+y)$ for $x, y \in \{0,1\}^k$. Then $G = A/2^k$.

---

[8] For $x \in \{0,1\}^k$, define unitary $U_{c^x} = \sum_{i\in[d]} |s_i\rangle\langle s_i| \otimes X^{(Mx)_i}$, where X is the NOT-gate, so $X^{(Mx)_i}|b\rangle = |b + (Mx)_i\rangle$ for $b \in \{0,1\}$. The ensemble $\mathcal{E}$ is generated by applying $\{U_{c^x}\}_{x\in\{0,1\}^k}$ to $|\varphi\rangle = \frac{1}{\sqrt{d}}\sum_{(i,b)\in[d]\times\{0,1\}} \sqrt{\frac{1}{2} + \frac{1}{2}(-1)^b\alpha}|s_i,b\rangle$. Since the 2^k codewords of the $[d,k,r]_2$ code form a linear subspace, $\{U_{c^x}\}_{x\in\{0,1\}^k}$ is an Abelian group. From the discussion in Section 2.6, we conclude that the PGM is the optimal measurement for this state identification problem.

Note that $T \leq d/(100e^3\varepsilon^2) \leq d/(e^3\beta)$ (the first inequality is by assumption and the second inequality follows for $\varepsilon \leq 1/10$ and $\beta \leq 1$). Since we assumed $T \leq d/(100e^3\varepsilon^2)$, we can use Theorem 17 (by choosing $m = d$ and $\beta = 1 - \sqrt{1-100\varepsilon^2}$) to upper bound the success probability of identifying the states in the ensemble $\mathcal{E}$:

$$\begin{aligned}
P^{PGM}(\mathcal{E}) &= \sum_{x\in\{0,1\}^k} \sqrt{G}(x,x)^2 \\
&= \frac{1}{2^k} \sum_{x\in\{0,1\}^k} \sqrt{A}(x,x)^2 && \text{(since } G = A/2^k\text{)} \\
&\leq \frac{4e}{2^k}\left(1-\frac{\beta}{2}\right)^T e^{22T^2\beta^2/d+2\sqrt{Td\beta}} && \text{(using Theorem 17)} \\
&\leq \frac{4e}{2^k}\left(1-\frac{\beta}{2}\right)^T e^{220000T^2\varepsilon^4/d+20\sqrt{Td\varepsilon^2}} && \text{(using } \beta = 1-\sqrt{1-100\varepsilon^2} \leq 100\varepsilon^2\text{)} \\
&\leq \frac{4e}{2^k}\left(1-25\varepsilon^2\right)^T e^{220000T^2\varepsilon^4/d+20\sqrt{Td\varepsilon^2}} && \text{(using } \sqrt{1-100\varepsilon^2} \leq 1-50\varepsilon^2\text{)} \\
&\leq \frac{4e}{e^{k\ln 2+25T\varepsilon^2}} e^{220000T^2\varepsilon^4/d+20\sqrt{Td\varepsilon^2}}. && \text{(using } (1-x)^t \leq e^{-xt} \text{ for } x,t \geq 0\text{)}
\end{aligned}$$

The lemma follows by observing that $k \geq d/4$. ◀
◀

4.3 Additional results

In this section we mention two additional results that can also be obtained using Theorem 17.

4.3.1 Quantum PAC sample complexity under random classification noise

In the theorem below, we show a lower bound on the quantum PAC sample complexity under the random classification noise model with noise rate η. Recall that in this model, for every $c \in \mathcal{C}$ and distribution D, $\varepsilon, \delta > 0$, given access to copies of the η-noisy state,

$$\sum_{x\in\{0,1\}^n} \sqrt{(1-\eta)D(x)}|x, c(x)\rangle + \sqrt{\eta D(x)}|x, 1-c(x)\rangle,$$

a (ε,δ)-PAC quantum learner is required to output an hypothesis h such that $\mathrm{err}_D(c,h) \leq \varepsilon$ with probability at least $1-\delta$.

▶ **Theorem 26.** *Let $\mathcal{C}$ be a concept class with VC-dim$(\mathcal{C}) = d+1$, for sufficiently large d. Then for every $\delta \in (0,1/2)$, $\varepsilon \in (0,1/20)$ and $\eta \in (0,1/2)$, every (ε,δ)-PAC quantum learner for $\mathcal{C}$ in the PAC setting with random classification noise rate η, has sample complexity $\Omega\left(\frac{d}{(1-2\eta)^2\varepsilon} + \frac{\log(1/\delta)}{(1-2\eta)^2\varepsilon}\right)$.*

One can use exactly the same proof technique as in Lemma 10 and Theorem 22 to prove this, with only the additional inequality $1-2\sqrt{\eta(1-\eta)} \leq (1-2\eta)^2$, which holds for $\eta \leq 1/2$. We omit the details of the calculation.

4.3.2 Distinguishing codeword states

Ashley Montanaro (personal communication) alerted us to the following interesting special case of our PGM-based result.

Consider an $[n,k,d]_2$ linear code $\{Mx : x \in \{0,1\}^k\}$, where $M \in \mathbb{F}_2^{n\times k}$ is the rank-k generator matrix of the code, $k = \Omega(n)$, and distinct codewords have Hamming distance at least d.[9] For every $x \in \{0,1\}^k$, define a *codeword state* $|\psi_x\rangle = \frac{1}{\sqrt{n}} \sum_{i\in[n]} |i,(Mx)_i\rangle$. These states form an example of a *quantum fingerprinting* scheme [19]: 2^k states whose pairwise inner products are bounded away from 1. How many copies do we need to identify one such fingerprint?

Let $\mathcal{E} = \{(2^{-k}, |\psi_x\rangle) : x \in \{0,1\}^k\}$ be an ensemble of codeword states. Consider the following task: given T copies of an unknown state drawn uniformly from $\mathcal{E}$, we are required to identify the state with probability $\geq 4/5$. From Holevo's theorem one can easily obtain a lower bound of $T = \Omega(k/\log n)$ copies, since the learner should obtain $\Omega(k)$ bits of information (i.e., identify k-bit string x with probability $\geq 4/5$), while each copy of the codeword state gives at most $\log n$ bits of information. In the theorem below, we improve that $\Omega(k/\log n)$ to the optimal $\Omega(k)$ for constant-rate codes.

▶ **Theorem 27.** *Let $\mathcal{E} = \{|\psi_x\rangle = \frac{1}{\sqrt{n}} \sum_{i\in[n]} |i,(Mx)_i\rangle : x \in \{0,1\}^k\}$, where $M \in \mathbb{F}_2^{n\times k}$ is the generator matrix of an $[n,k,d]_2$ linear code with $k = \Omega(n)$. Then $\Omega(k)$ copies of an unknown state from $\mathcal{E}$ (drawn uniformly at random) are necessary to be able to identify that state with probability at least $4/5$.*

One can use exactly the proof technique of Theorem 22 to prove the theorem. Suppose we are given T copies of the unknown codeword state. Assume $T \leq n$, since otherwise $T \geq n \geq \sqrt{kn}$ and the theorem follows. Observe that the Gram matrix G for $\mathcal{E}' = \{2^{-k/2}|\psi_x\rangle^{\otimes T} : x \in \{0,1\}^k\}$ can be written as $G(x,y) = \frac{1}{2^k}\left(1 - \frac{|M(x+y)|}{n}\right)^T$ for $x,y \in \{0,1\}^k$. Using Theorem 17 (choosing $\beta = 1$ and $m = n$) to upper bound the success probability of successfully identifying the states in the ensemble $\mathcal{E}$ using the PGM, we obtain

$$P^{PGM}(\mathcal{E}) \leq \frac{4e}{2^{k+T}} e^{22T^2/n + 2\sqrt{Tn}}.$$

As in the proof of Theorem 22, this implies the lower bound of Theorem 27. We omit the details of the calculation.

5 Conclusion

The main result of this paper is that quantum examples give no significant improvement over the usual random examples in passive, distribution-independent settings. Of course, these negative results do not mean that quantum machine learning is useless. In our introduction we already mentioned improvements from quantum examples for learning under the uniform distribution; improvements from using quantum membership queries; and improvements in time complexity based on quantum algorithms like Grover's and HHL. Quantum machine learning is still in its infancy, and we hope for many more positive results.

We end by identifying a number of open questions for future work:

- We gave lower bounds on sample complexity for the rather benign random classification noise. What about other noise models, such a *malicious* noise?
- What is the quantum sample complexity for learning concepts whose range is $[k]$ rather than $\{0,1\}$, for some $k > 2$? Even the *classical* sample complexity is not fully determined yet [45, Section 29.2].

[9] Note that throughout this paper $\mathcal{C}$ was a concept class in $\{0,1\}^n$ and d was the VC dimension of $\mathcal{C}$. The use of n, d in this section has been changed to conform to the convention in coding theory.

- Classically, it is still an open question whether the $\log(1/\varepsilon)$-factor in the upper bound of [17] for (ε, δ)-*proper* PAC learning is necessary. A weaker result (possibly easier to prove) would be to give a (ε, δ)-*quantum* proper PAC learner without this $\log(1/\varepsilon)$-factor.
- In the introduction we mentioned a few examples of learning under the *uniform* distribution where quantum examples are significantly more powerful than classical examples. Can we find more such examples of quantum improvements in sample complexity in fixed-distribution settings?
- Can we find more examples of quantum speed-up in *time* complexity of learning, for example for learning depth-3 or even constant-depth circuits?

Acknowledgements. We thank Shalev Ben-David, Dmitry Gavinsky, Robin Kothari, Nishant Mehta, Ashley Montanaro, Henry Yuen for helpful comments and pointers to the literature. We also thank Ashley Montanaro for suggesting the additional remark in Section 4.3.2.

References

1 S. Aaronson. The learnability of quantum states. *Proceedings of the Royal Society of London*, 463(2088), 2007. quant-ph/0608142.

2 S. Aaronson. Quantum machine learning algorithms: Read the fine print. *Nature Physics*, 11(4):291–293, April 2015.

3 E. Aïmeur, G. Brassard, and S. Gambs. Machine learning in a quantum world. In *Proceedings of Advances in Artificial Intelligence, 19th Conference of the Canadian Society for Computational Studies of Intelligence*, volume 4013 of *Lecture Notes in Artificial Intelligence*, pages 431–442, 2006.

4 E. Aïmeur, G. Brassard, and S. Gambs. Quantum speed-up for unsupervised learning. *Machine Learning*, 90(2):261–287, 2013.

5 A. Ambainis and A. Montanaro. Quantum algorithms for search with wildcards and combinatorial group testing. *Quantum Information & Computation*, 14(5-6):439–453, 2014. arXiv:1210.1148.

6 D. Angluin and P. Laird. Learning from noisy examples. *Machine Learning*, 2(4):343–370, 1988.

7 M. Anthony and P. L. Bartlett. *Neural network learning: Theoretical foundations*. Cambridge University Press, 2009.

8 B. Apolloni and C. Gentile. Sample size lower bounds in PAC learning by algorithmic complexity theory. *Theoretical Computer Science*, 209:141–162, 1998.

9 S. Arunachalam and R. de Wolf. A survey of quantum learning theory, 2017. To appear as Computational Complexity Column in SIGACT News, June 2017. Preprint at arxiv:1606.08920.

10 A. Atıcı and R. Servedio. Improved bounds on quantum learning algorithms. *Quantum Information Processing*, 4(5):355–386, 2005. quant-ph/0411140.

11 A. Atıcı and R. Servedio. Quantum algorithms for learning and testing juntas. *Quantum Information Processing*, 6(5):323–348, 2009. arXiv:0707.3479.

12 J. Audibert. Fast learning rates in statistical inference through aggregation, 2008. Research Report 06-20, Certis – Ecole des Ponts. math/0703854.

13 J. Audibert. Fast learning rates in statistical inference through aggregation. *The Annals of Statistics*, 37(4):1591–1646, 2009. arXiv:0909.1468v1.

14 D. Bacon, A. Childs, and W. van Dam. Optimal measurements for the dihedral hidden subgroup problem. *Chicago Journal of Theoretical Computer Science*, 2006. Earlier version in FOCS'05. quant-ph/0504083.

15 H. Barnum and E. Knill. Reversing quantum dynamics with near-optimal quantum and classical fidelity. *Journal of Mathematical Physics*, 43:2097–2106, 2002. quant-ph/0004088.

16 E. Bernstein and U. Vazirani. Quantum complexity theory. *SIAM Journal on Computing*, 26(5):1411–1473, 1997. Earlier version in STOC'93.

17 A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM*, 36(4):929–965, 1989.

18 N. H. Bshouty and J. C. Jackson. Learning DNF over the uniform distribution using a quantum example oracle. *SIAM Journal on Computing*, 28(3):1136–1153, 1999. Earlier version in COLT'95.

19 H. Buhrman, R. Cleve, J. Watrous, and R. de Wolf. Quantum fingerprinting. *Physical Review Letters*, 87(16), 2001. quant-ph/0102001.

20 A. Daniely and S. Shalev-Shwartz. Complexity theoretic limitations on learning DNF's. In *Proceedings of the 29th Conference on Learning Theory (COLT'16)*, 2016.

21 A. Ehrenfeucht, D. Haussler, M. J. Kearns, and L. G. Valiant. A general lower bound on the number of examples needed for learning. *Information and Computation*, 82(3):247–261, 1989. Earlier version in COLT'98.

22 Y. C. Eldar and G. D. Forney Jr. On quantum detection and the square-root measurement. *IEEE Transactions and Information Theory*, 47(3):858–872, 2001. quant-ph/0005132.

23 Y. C. Eldar, A. Megretski, and G. C. Verghese. Designing optimal quantum detectors via semidefinite programming. *IEEE Transactions Information Theory*, 49(4):1007–1012, 2003. quant-ph/0205178.

24 D. Gavinsky. Quantum predictive learning and communication complexity with single input. *Quantum Information and Computation*, 12(7-8):575–588, 2012. Earlier version in COLT'10. arXiv:0812.3429.

25 C. Gentile and D. P. Helmbold. Improved lower bounds for learning from noisy examples: An information-theoretic approach. *Information and Computation*, 166:133–155, 2001.

26 L. K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of 28th ACM STOC*, pages 212–219, 1996. quant-ph/9605043.

27 S. Hanneke. The optimal sample complexity of PAC learning. *Journal of Machine Learning Research*, 17(38):1–15, 2016. arXiv:1507.00473.

28 A. Harrow, A. Hassidim, and S. Lloyd. Quantum algorithm for solving linear systems of equations. *Physical Review Letters*, 103(15):150502, 2009. arXiv:0811.3171.

29 P. Hausladen, R. Jozsa, B. Schumacher, M. Westmoreland, and W. K. Wootters. Classical information capacity of a quantum channel. *Physical Review A*, 54:1869–1876, 1996.

30 P. Hausladen and W. K. Wootters. A 'pretty good' measurement for distinguishing quantum states. *Journal of Modern Optics*, 41:2385–2390, 1994.

31 D. Haussler. Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Information and Computation*, 100(1):78–150, 1992.

32 M. Hunziker, D. A. Meyer, J. Park, J. Pommersheim, and M. Rothstein. The geometry of quantum learning. *Quantum Information Processing*, 9(3):321–341, 2010. quant-ph/0309059.

33 J. C. Jackson. An efficient membership-query algorithm for learning DNF with respect to the uniform distribution. *Journal of Computer and System Sciences*, 55(3):414–440, 1997. Earlier version in FOCS'94.

34 J. C. Jackson, C. Tamon, and T. Yamakami. Quantum DNF learnability revisited. In *Proceedings of 8th COCOON*, pages 595–604, 2002. quant-ph/0202066.

35 R. Jain and S. Zhang. New bounds on classical and quantum one-way communication complexity. *Theoretical Computer Science*, 410(26):2463–2477, 2009. arXiv:0802.4101.

36 M. J. Kearns, R. E. Schapire, and L. Sellie. Toward efficient agnostic learning. *Machine Learning*, 17(2-3):115–141, 1994. Earlier version in COLT'92.

37 M. J. Kearns and L. G. Valiant. Cryptographic limitations on learning Boolean formulae and finite automata. *Journal of the ACM*, 41(1):67–95, 1994.

38 M. J. Kearns and U. V. Vazirani. *An introduction to computational learning theory*. MIT Press, 1994.

39 A. Kontorovich and I. Pinelis. Exact lower bounds for the agnostic probably-approximately-correct (PAC) machine learning model, 2016. Preprint at arxiv:1606.08920.

40 R. Kothari. An optimal quantum algorithm for the oracle identification problem. In *31st International Symposium on Theoretical Aspects of Computer Science (STACS 2014)*, pages 482–493, 2014. arXiv:1311.7685.

41 A. Montanaro. On the distinguishability of random quantum states. *Communications in Mathematical Physics*, 273(3):619–636, 2007. quant-ph/0607011.

42 A. Montanaro. The quantum query complexity of learning multilinear polynomials. *Information Processing Letters*, 112(11):438–442, 2012. arXiv:1105.3310.

43 R. O'Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2014.

44 R. Servedio and S. Gortler. Equivalences and separations between quantum and classical learnability. *SIAM Journal on Computing*, 33(5):1067–1092, 2004. Combines earlier papers from ICALP'01 and CCC'01. quant-ph/0007036.

45 S. Shalev-Shwartz and S. Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge University Press, 2014.

46 H. U. Simon. General bounds on the number of examples needed for learning probabilistic concepts. *Journal of Computer and System Sciences*, 52(2):239–254, 1996. Earlier version in COLT'93.

47 H. U. Simon. An almost optimal PAC algorithm. In *Proceedings of the 28th Conference on Learning Theory (COLT)*, pages 1552–1563, 2015.

48 M. Talagrand. Sharper bounds for Gaussian and empirical processes. *The Annals of Probability*, pages 28–76, 1994.

49 L. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

50 V. Vapnik and A. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability & Its Applications*, 16(2):264–280, 1971.

51 V. N. Vapnik and A. Ya. Chervonenkis. *Theory of Pattern Recognition*. Nauka, 1974. In Russian.

52 K. A. Verbeurgt. Learning DNF under the uniform distribution in quasi-polynomial time. In *Proceedings of the 3rd Annual Workshop on Computational Learning Theory (COLT'90)*, pages 314–326, 1990.

53 N. Wiebe, A. Kapoor, and K. M. Svore. Quantum deep learning, 2014. Preprint at arXiv:1412.3489.

54 N. Wiebe, A. Kapoor, and K. M. Svore. Quantum perceptron models, 2016. Preprint at arXiv:1602.04799.

55 C. Zhang. An improved lower bound on query complexity for quantum PAC learning. *Information Processing Letters*, 111(1):40–45, 2010.

Settling the Query Complexity of Non-Adaptive Junta Testing*

Xi Chen[1], Rocco A. Servedio[2], Li-Yang Tan[3], Erik Waingarten[4], and Jinyu Xie[5]

1 **Columbia University, New York, NY, USA**
`xichen@cs.columbia.edu`
2 **Columbia University,New York, NY, USA**
`rocco@cs.columbia.edu`
3 **Toyota Technological Institute, Chicago, IL, USA**
`liyang@cs.columbia.edu`
4 **Columbia University, New York, NY, USA**
`eaw@cs.columbia.edu`
5 **Columbia University, New York, NY, USA**
`jinyu@cs.columbia.edu`

Abstract

We prove that any non-adaptive algorithm that tests whether an unknown Boolean function $f\colon \{0,1\}^n \to \{0,1\}$ is a k-junta or ϵ-far from every k-junta must make $\widetilde{\Omega}(k^{3/2}/\epsilon)$ many queries for a wide range of parameters k and ϵ. Our result dramatically improves previous lower bounds from [12, 38], and is essentially optimal given Blais's non-adaptive junta tester from [7], which makes $\widetilde{O}(k^{3/2})/\epsilon$ queries. Combined with the adaptive tester of [8] which makes $O(k \log k + k/\epsilon)$ queries, our result shows that adaptivity enables polynomial savings in query complexity for junta testing.

1998 ACM Subject Classification F.2 Analysis of Algorithms and Problem Complexity

Keywords and phrases property testing, juntas, query complexity

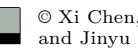
Digital Object Identifier 10.4230/LIPIcs.CCC.2017.26

1 Introduction

This paper is concerned with the power of adaptivity in property testing, specifically property testing of Boolean functions. At a high level, a property tester for Boolean functions is a randomized algorithm which, given black-box query access to an unknown and arbitrary Boolean function $f\colon \{0,1\}^n \to \{0,1\}$, aims to distinguish between the case that f has some particular property of interest versus the case that f is far in Hamming distance from every Boolean function satisfying the property. The main goals in the study of property testing algorithms are to develop testers that make *as few queries* as possible, and to establish lower bounds matching these query-efficient algorithms. Property testing has by now been studied for many different types of Boolean functions, including linear functions and low-degree polynomials over $GF(2)$ [11, 2, 6], literals, conjunctions, s-term monotone and non-monotone

* X.C. and J.X. were supported by NSF awards CCF-1149257 and CCF-1423100.
R.A.S. was supported by NSF awards CCF-1420349 and CCF-1563155.
L.-Y.T. was supported by NSF award CCF-1563122.
E.W. was supported by the NSF Graduate Research Fellowship under Grant No. DGE-16-44869.

32nd Computational Complexity Conference (CCC 2017).
Editor: Ryan O'Donnell; Article No. 26; pp. 26:1–26:19



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

DNFs [32, 18], monotone and unate functions [23, 20, 13, 16, 15, 27, 4, 28, 14, 3], various types of linear threshold functions [30, 31, 9], size-s decision trees and s-sparse $GF(2)$ polynomials and parities [18, 9, 10], functions with sparse or low-degree Fourier spectrum [24], and much more. See e.g. [33, 34, 22] for some fairly recent broad overviews of property testing research.

In this work we consider the property of being a *k-junta*, which is one of the earliest and most intensively studied properties in the Boolean function property testing literature. Recall that f is a k-junta if it has at most k relevant variables, i.e., there exist k distinct indices $i_1, \ldots, i_k$ and a k-variable function $g\colon \{0,1\}^k \to \{0,1\}$ such that $f(x) = g(x_{i_1}, \ldots, x_{i_k})$ for all $x \in \{0,1\}^n$. Given $k = k(n)\colon \mathbb{N} \to \mathbb{N}$ and $\epsilon = \epsilon(n)\colon \mathbb{N} \to \mathbb{R}_{>0}$, we say an algorithm which has black-box access to an unknown and arbitrary $f\colon \{0,1\}^n \to \{0,1\}$ is an *ϵ-tester* or *ϵ-testing algorithm for k-juntas* if it accepts with probability at least $5/6$ when f is a $k(n)$-junta and rejects with probability at least $5/6$ when f is $\epsilon(n)$-far from all $k(n)$-juntas (meaning that f disagrees with any $k(n)$-junta g on at least $\epsilon(n) \cdot 2^n$ many inputs).

Property testers come in two flavors, adaptive and non-adaptive. An adaptive tester receives the value of f on its i-th query string before deciding on its $(i+1)$-st query string, while a non-adaptive tester selects all of its query strings before receiving the value of f on any of them. Note that non-adaptive testers can evaluate all of their queries in one parallel stage of execution, while this is in general not possible for adaptive testers. This means that if evaluating a query is very time-consuming, non-adaptive algorithms may sometimes be preferable to adaptive algorithms even if they require more queries. For this and other reasons, it is of interest to understand when, and to what extent, adaptive algorithms can use fewer queries than non-adaptive algorithms (see [36, 35] for examples of property testing problems where indeed adaptive algorithms are provably more query-efficient than non-adaptive ones).

The query complexity of adaptive junta testing algorithms is at this point well understood. In [17] Chockler and Gutfreund showed that even adaptive testers require $\Omega(k)$ queries to distinguish k-juntas from random functions on $k+1$ variables, which are easily seen to be constant-far from k-juntas. Blais [8] gave an adaptive junta testing algorithm that uses only $O(k \log k + k/\epsilon)$ queries, which is optimal (for constant ϵ) up to a multiplicative factor of $O(\log k)$.

Prior to the current work, the picture was significantly less clear for non-adaptive junta testing. In the first work on junta testing, Fischer et al. [19] gave a non-adaptive tester that makes $O(k^2(\log k)^2/\epsilon)$ queries. This was improved by Blais [7] with a non-adaptive tester that uses only $O(k^{3/2}(\log k)^3/\epsilon)$ queries. On the lower bounds side, [7] also showed that for all $\epsilon \geq k/2^k$, any non-adaptive algorithm for ϵ-testing k-juntas must make $\Omega\left(k/(\epsilon \log(k/\epsilon))\right)$ queries. Buhrman et al. [12] gave an $\Omega(k \log k)$ lower bound (for constant ϵ) for non-adaptively testing whether a function f is a size-k parity; their argument also yields an $\Omega(k \log k)$ lower bound (for constant ϵ) for non-adaptively ϵ-testing k-juntas. More recently, [38] obtained a new lower bound for non-adaptive junta testing that is incomparable to both the [7] and the [12] lower bounds. They showed that for all $\epsilon : k^{-o_k(1)} \leq \epsilon \leq o_k(1)$, any non-adaptive ϵ-tester for k-juntas must make

$$\Omega\left(\frac{k \log k}{\epsilon^c \log(\log(k)/\epsilon^c)}\right)$$

many queries, where c is any absolute constant less than 1. For certain restricted values of ϵ such as $\epsilon = 1/\log k$, this lower bound is larger than the $O(k/\epsilon + k \log k)$ upper bound for [8]'s adaptive algorithm, so the [38] lower bound shows that in some restricted settings, adaptive junta testers can outperform non-adaptive ones. However, the difference in performance is quite small, at most a $o(\log k)$ factor. We further note that all of the lower bounds [7, 12, 38]

are of the form $\widetilde{\Omega}(k)$ for constant ϵ, and hence rather far from the $\widetilde{O}(k^{3/2})/\epsilon$ upper bound of [7].

1.1 Our results

The main result of the paper is the following theorem:

▶ **Theorem 1.** *Let $\alpha \in (0.5, 1)$ be an absolute constant. Let $k = k(n)\colon \mathbb{N} \to \mathbb{N}$ and $\epsilon = \epsilon(n)\colon \mathbb{N} \to \mathbb{R}_{>0}$ be two functions that satisfy $k(n) \leq \alpha n$ and $2^{-n} \leq \epsilon(n) \leq 1/6$ for all sufficiently large n. Then any non-adaptive ϵ-tester for k-juntas must make $\widetilde{\Omega}(k^{3/2}/\epsilon)$ many queries.*

Together with the $\widetilde{O}(k^{3/2})/\epsilon$ non-adaptive upper bound from [7], Theorem 1 settles the query complexity of non-adaptive junta testing up to poly-logarithmic factors.

1.2 High-level overview of our approach

Our lower bound approach differs significantly from previous work. Buhrman et al. [12] leveraged the connection between communication complexity lower bounds and property testing lower bounds that was established in the work of [9] and applied an $\Omega(k \log k)$ lower bound on the one-way communication complexity of k-disjointness to establish their lower bound. Both [7] and [38] are based on edge-isoperimetry results for the Boolean hypercube (the edge-isoperimetric inequality of Harper [25], Bernstein [5], Lindsey [29], and Hart [26] in the case of [7], and a slight extension of a result of Frankl [21] in [38]). In contrast, our lower bound argument takes a very different approach; it consists of a sequence of careful reductions, and employs an *upper* bound on the total variation distance between two Binomial distributions (see Claim 15).

Below we provide a high level overview of the proof of the lower bound given by Theorem 1. First, it is not difficult to show that Theorem 1 is a consequence of the following more specific lower bound for the case where $k = \alpha n$:

▶ **Theorem 2.** *Let $\alpha \in (0.5, 1)$ be an absolute constant. Let $k = k(n)\colon \mathbb{N} \to \mathbb{N}$ and $\epsilon = \epsilon(n)\colon \mathbb{N} \to \mathbb{R}_{>0}$ be two functions that satisfy $k(n) = \alpha n$ and $2^{-(2\alpha-1)n/2} \leq \epsilon(n) \leq 1/6$ for sufficiently large n. Then any non-adaptive ϵ-tester for k-juntas must make $\widetilde{\Omega}(n^{3/2}/\epsilon)$ many queries.*

See Appendix A for the proof that Theorem 2 implies Theorem 1.

We now provide a sketch of how Theorem 2 is proved. It may be convenient for the reader, on the first reading, to consider $\alpha = 3/4$ and to think of ϵ as being a small constant such as 0.01.

Fix a sufficiently large n. Let $k = \alpha n$ and $\epsilon = \epsilon(n)$ with ϵ satisfying the condition in Theorem 2. We proceed by Yao's principle and prove lower bounds for deterministic non-adaptive algorithms which receive inputs drawn from one of two probability distributions, $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$, over n-variable Boolean functions. The distributions $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$ are designed so that a Boolean function $\boldsymbol{f} \leftarrow \mathcal{D}_{\text{yes}}$ is a k-junta with probability $1 - o(1)$ and $\boldsymbol{f} \leftarrow \mathcal{D}_{\text{no}}$ is ϵ-far from every k-junta with probability $1 - o(1)$. In Section 2 we define $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$, and establish the above properties. By Yao's principle, it then suffices to show that any q-query non-adaptive deterministic algorithm (i.e., any set of q queries) that succeeds in distinguishing them must have $q = \widetilde{\Omega}(n^{3/2}/\epsilon)$.

This lower bound proof consists of two components:

1. A reduction from a simple algorithmic task called Set-Size-Set-Queries (SSSQ for short), which we discuss informally later in this subsection and we define formally in Section 3. This reduction implies that the non-adaptive deterministic query complexity of distinguishing $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$ is at least as large as that of SSSQ.
2. A lower bound of $\widetilde{\Omega}(n^{3/2}/\epsilon)$ for the query complexity of SSSQ.

Having outlined the formal structure of our proof, let us give some intuition which may hopefully be helpful in motivating our construction and reduction. Our yes-functions and no-functions have very similar structure to each other, but are constructed with slightly different parameter settings. The first step in drawing a random function from $\mathcal{D}_{\text{yes}}$ is choosing a uniform random subset $\mathbf{M}$ of $\Theta(n)$ "addressing" variables from $x_1, \ldots, x_n$. A random subset $\mathbf{A}$ of the complementary variables $\overline{\mathbf{M}}$ is also selected, and for each assignment to the variables in $\mathbf{M}$ (let us denote such an assignment by i), there is an independent random function $\boldsymbol{h}_i$ over a randomly selected subset $\mathbf{S}_i$ of the variables in $\mathbf{A}$. A random function from $\mathcal{D}_{\text{no}}$ is constructed in the same way, except that now the random subset $\mathbf{A}$ is chosen to be slightly larger than in the yes-case. This disparity in the size of $\mathbf{A}$ between the two cases causes random functions from $\mathcal{D}_{\text{yes}}$ to almost always be k-juntas and random functions from $\mathcal{D}_{\text{no}}$ to almost always be far from k-juntas.

An intuitive explanation of why this construction is amenable to a lower bound for non-adaptive algorithms is as follows. Intuitively, for an algorithm to determine that it is interacting with (say) a random no-function rather than a random yes-function, it must determine that the subset $\mathbf{A}$ is larger than it should be in the yes-case. Since the set $\mathbf{M}$ of $\Theta(n)$ many "addressing" variables is selected randomly, if a non-adaptive algorithm uses two query strings x, x' that differ in more than a few coordinates, it is very likely that they will correspond to two different random functions $\boldsymbol{h}_i, \boldsymbol{h}_{i'}$. Hence every pair of query strings x, x' that correspond to the same $\boldsymbol{h}_i$ can differ only in a few coordinates in $\overline{\mathbf{M}}$, with high probability, which significantly limits the power of a non-adaptive algorithm in distinguishing $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$ no matter which set of query strings it picks. This makes it possible for us to reduce from the SSSQ problem to the problem of distinguishing $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$ at the price of only a small quantitative cost in query complexity, see Section 4.

At a high level, the SSSQ task involves distinguishing whether or not a hidden set (corresponding to $\mathbf{A}$) is "large." An algorithm for this task can only access certain random bits, whose biases are determined by the hidden set and whose exact distribution is inspired by the exact definition of the random functions $\boldsymbol{h}_i$ over the random subsets $\mathbf{S}_i$. Although SSSQ is an artificial problem, it is much easier to work with compared to the original problem of distinguishing $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$. In particular, we give a reduction from an even simpler algorithmic task called Set-Size-Element-Queries (SSEQ for short) to SSSQ (see Section 5.1) and the query complexity lower bound for SSSQ follows directly from the lower bound for SSEQ presented in Section 5.2.

Let us give a high-level description of the SSEQ task to provide some intuition for how we prove a query lower bound on it. Roughly speaking, in this task an oracle holds an unknown and random subset $\mathbf{A}$ of $[m]$ (here $m = \Theta(n)$) which is either "small" (size roughly $m/2$) or "large" (size roughly $m/2 + \Theta(\sqrt{n} \cdot \log n)$), and the task is to determine whether $\mathbf{A}$ is small or large. The algorithm may repeatedly query the oracle by providing it, at the j-th query, with an element $i_j \in [m]$; if $i_j \notin \mathbf{A}$ then the oracle responds "0" with probability 1, and if $i_j \in \mathbf{A}$ then the oracle responds "1" with probability $\epsilon/\sqrt{n}$ and "0" otherwise. Intuitively, the only way for an algorithm to determine that the unknown set $\mathbf{A}$ is (say) large, is to determine that the fraction of elements of $[m]$ that belong to $\mathbf{A}$ is $1/2 + \Theta(\log n/\sqrt{n})$ rather than $1/2$; this in turn intuitively requires sampling $\Omega(n/\log^2 n)$ many random elements of

$[m]$ and for each one ascertaining with high confidence whether or not it belongs to $\mathbf{A}$. But the nature of the oracle access described above for SSEQ is such that for any given $i \in [m]$, at least $\Omega(\sqrt{n}/\epsilon)$ many repeated queries to the oracle on input i are required in order to reach even a modest level of confidence as to whether or not $i \in \mathbf{A}$. As alluded to earlier, the formal argument establishing our lower bound on the query complexity of SSEQ relies on an upper bound on the total variation distance between two Binomial distributions.

1.3 Organization and Notation

We start with the definitions of $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$ as well as proofs of their properties in Section 2. We then introduce SSSQ in Section 3, and give a reduction from SSSQ to the problem of distinguishing $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$ in Section 4. More formally, we show that any non-adaptive deterministic algorithm that distinguishes $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$ can be used to solve SSSQ with only an $O(\log n)$ factor loss in the query complexity. Finally, we prove in Section 5 a lower bound for the query complexity of SSSQ. Theorem 2 then follows by combining this lower bound with the reduction in Section 4.

We use boldfaced letters such as $\boldsymbol{f}, \mathbf{A}, \mathbf{S}$ to denote random variables. Given a string $x \in \{0,1\}^n$ and $\ell \in [n]$, we write $x^{(\ell)}$ to denote the string obtained from x by flipping the ℓ-th coordinate. An *edge* along the ℓth direction in $\{0,1\}^n$ is a pair (x, y) of strings with $y = x^{(\ell)}$. We say an edge (x, y) is *bichromatic with respect to a function* f (or simply f-bichromatic) if $f(x) \neq f(y)$. Given $x \in \{0,1\}^n$ and $S \subseteq [n]$, we use $x_{|S} \in \{0,1\}^S$ to denote the projection of x on S.

2 The $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$ distributions

Let $\alpha \in (0.5, 1)$ be an absolute constant. Let n be a sufficiently large integer, with $k = \alpha n$, and let ϵ be the distance parameter that satisfies

$$2^{-(2\alpha-1)n/2} \leq \epsilon \leq 1/6. \tag{1}$$

In this section we describe a pair of probability distributions $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$ supported over Boolean functions $f\colon \{0,1\}^n \to \{0,1\}$. We then show that $\boldsymbol{f} \leftarrow \mathcal{D}_{\text{yes}}$ is a k-junta with probability $1 - o(1)$, and that $\boldsymbol{f} \leftarrow \mathcal{D}_{\text{no}}$ is ϵ-far from being a k-junta with probability $1 - o(1)$.

We start with some parameters settings.

Define

$$\delta \stackrel{\text{def}}{=} 1 - \alpha \in (0, 0.5), \qquad p \stackrel{\text{def}}{=} \frac{1}{2}, \qquad q \stackrel{\text{def}}{=} \frac{1}{2} + \frac{\log n}{\sqrt{n}},$$

$$m \stackrel{\text{def}}{=} 2\delta n + \delta\sqrt{n}\log n, \qquad t \stackrel{\text{def}}{=} n - m = (2\alpha - 1)n - \delta\sqrt{n}\log n, \qquad N \stackrel{\text{def}}{=} 2^t.$$

A function $\boldsymbol{f} \leftarrow \mathcal{D}_{\text{yes}}$ is drawn according to the following randomized procedure:

1. Sample a random subset $\mathbf{M} \subset [n]$ of size t. Let $\boldsymbol{\Gamma} = \Gamma_{\mathbf{M}} : \{0,1\}^n \to [N]$ be the function that maps $x \in \{0,1\}^n$ to the integer encoded by $x_{|\mathbf{M}}$ in binary plus one. Note that $|\overline{\mathbf{M}}| = n - t = m$.
2. Sample an $\mathbf{A} \subseteq \overline{\mathbf{M}}$ by including each element of $\overline{\mathbf{M}}$ in $\mathbf{A}$ independently with probability p.
3. Sample independently a sequence of N random subsets $\mathbf{S} = (\mathbf{S}_i : i \in [N])$ of $\mathbf{A}$ as follows: for each $i \in [N]$, each element of $\mathbf{A}$ is included in $\mathbf{S}_i$ independently with probability $\epsilon/\sqrt{n}$. Next we sample a sequence of N functions $\mathbf{H} = (\boldsymbol{h}_i : i \in [N])$, by letting $\boldsymbol{h}_i : \{0,1\}^n \to \{0,1\}$ be a random function over the coordinates in $\mathbf{S}_i$, i.e., we sample an unbiased bit $\boldsymbol{z}_i(b)$ for each string $b \in \{0,1\}^{\mathbf{S}_i}$ independently and set $\boldsymbol{h}_i(x) = \boldsymbol{z}_i(x_{|\mathbf{S}_i})$.

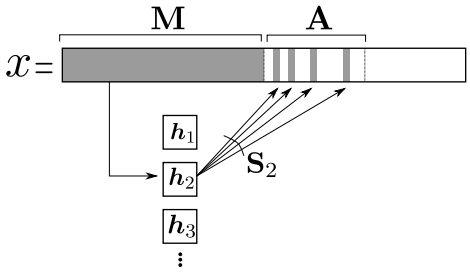


Figure 1 An example of how an input $x \in \{0,1\}^n$ is evaluated by $\boldsymbol{f} \sim \mathcal{D}_{\text{yes}}$ (or $\mathcal{D}_{\text{no}}$). The relevant variables of x are shaded gray. All variables in $\mathbf{M}$ index x into $\boldsymbol{h}_2$, which is a random function over the variables $\mathbf{S}_2$, which are sampled from $\mathbf{A}$ by including each with probability $\epsilon/\sqrt{n}$.

4. Finally, $\boldsymbol{f} = \boldsymbol{f}_{\mathbf{M},\mathbf{A},\mathbf{H}} \colon \{0,1\}^n \to \{0,1\}$ is defined using $\mathbf{M}, \mathbf{A}$ and $\mathbf{H}$ as follows:

   $$\boldsymbol{f}(x) = \boldsymbol{h}_{\Gamma_{\mathbf{M}}(x)}(x), \quad \text{for each } x \in \{0,1\}^n.$$

   In words, an input x is assigned the value $\boldsymbol{f}(x)$ as follows: according to the coordinates of x in the set $\mathbf{M}$ (which intuitively should be thought of as unknown), one of the N functions $\boldsymbol{h}_i$ (each of which is, intuitively, a random function over an unknown subset $\mathbf{S}_i$ of coordinates) is selected and evaluated on x's coordinates in $\mathbf{S}_i$. For intuition, we note that both $\mathbf{M}$ and $\overline{\mathbf{M}}$ will always be of size $\Theta(n)$, the size of $\mathbf{A}$ will almost always be $\Theta(n)$, and for a given $i \in [N]$ the expected size of $\mathbf{S}_i$ will typically be $\Theta(\epsilon\sqrt{n})$ (though the size of $\mathbf{S}_i$ may not be as highly concentrated as the other sets when ϵ is tiny).

A function $\boldsymbol{f} \leftarrow \mathcal{D}_{\text{no}}$ is generated using the same procedure except that $\mathbf{A}$ is a random subset of $\overline{\mathbf{M}}$ drawn by including each element of $\overline{\mathbf{M}}$ in $\mathbf{A}$ independently with probability q (instead of p). See Figure 1 for an example of how an input $x \in \{0,1\}^n$ is evaluated by $\boldsymbol{f} \sim \mathcal{D}_{\text{yes}}$ or $\mathcal{D}_{\text{no}}$.

2.1 Most functions drawn from $\mathcal{D}_{\text{yes}}$ are k-juntas

We first prove that $\boldsymbol{f} \leftarrow \mathcal{D}_{\text{yes}}$ is a k-junta with probability $1 - o(1)$.

▶ **Lemma 3.** *A function $\boldsymbol{f} \leftarrow \mathcal{D}_{yes}$ is a k-junta with probability $1 - o(1)$.*

Proof. By the definition of $\mathcal{D}_{\text{yes}}$, all the relevant variables of $\boldsymbol{f} \sim \mathcal{D}_{\text{yes}}$ belong to $\mathbf{M} \cup \mathbf{A}$. Note that $|\mathbf{M}| = t$. On the other hand, the expected size of $\mathbf{A}$ is $\delta n + \delta\sqrt{n}\log n/2$. By a Chernoff bound,

$$|\mathbf{A}| \leq \delta n + \frac{\delta\sqrt{n}\log n}{2} + \frac{\delta\sqrt{n}\log n}{4} < \delta n + \delta\sqrt{n}\log n$$

with probability $1 - o(1)$. When this happens we have $|\mathbf{M} \cup \mathbf{A}| < \alpha n = k$. ◀

2.2 Most functions drawn from $\mathcal{D}_{\text{no}}$ are ϵ-far from k-juntas

Next we prove that $\boldsymbol{f} \leftarrow \mathcal{D}_{\text{no}}$ is ϵ-far from any k-junta with probability $1 - o(1)$. The details of the argument are somewhat technical so we start by giving some high-level intuition, which is relatively simple. Since $q = p + \log(n)/\sqrt{n}$, a typical outcome of $\mathbf{A}$ drawn from $\mathcal{D}_{\text{no}}$ is slightly larger than a typical outcome drawn from $\mathcal{D}_{\text{yes}}$, and this difference causes almost every outcome of $|\mathbf{M} \cup \mathbf{A}|$ in $\mathcal{D}_{\text{no}}$ (with $\mathbf{M} \cup \mathbf{A}$ being the set of relevant variables

for $\boldsymbol{f} \leftarrow \mathcal{D}_{\text{no}}$) to be larger than k by at least $9\sqrt{n}$. As a result, the relevant variables of any k-junta must miss either (a) at least one variable from $\mathbf{M}$, or (b) at least $9\sqrt{n}$ variables from $\mathbf{A}$. Missing even a single variable from $\mathbf{M}$ causes the k-junta to be far from $\boldsymbol{f}$ (this is made precise in Claim 6 below). On the other hand, missing $9\sqrt{n}$ variables from $\mathbf{A}$ means that with probability at least $\Omega(\epsilon)$, at least one variable is missing from a typical $\mathbf{S}_i$ (recall that these are random $(\epsilon/\sqrt{n})$-dense subsets of $\mathbf{A}$). Because $\boldsymbol{h}_i$ is a random function over the variables in $\mathbf{S}_i$, missing even a single variable would lead to a constant fraction of error when $\boldsymbol{h}_i$ is the function determining the output of $\boldsymbol{f}$.

▶ **Lemma 4.** *A function $\boldsymbol{f} \leftarrow \mathcal{D}_{no}$ is ϵ-far from being a k-junta with probability $1 - o(1)$.*

Proof. Fix any subset $M \subset [n]$ of size t, and we consider $\boldsymbol{f} = \boldsymbol{f}_{M,\mathbf{A},\mathbf{H}}$ where $\mathbf{A}$ and $\mathbf{H}$ are sampled according to the procedure for $\mathcal{D}_{\text{no}}$. With probability $1 - o(1)$ over the choice of $\mathbf{A}$, we have

$$|\mathbf{A}| \geq qm - \frac{\delta\sqrt{n}\log n}{2} \geq \delta n + 2\delta\sqrt{n}\log n \quad \text{and} \quad |\mathbf{M} \cup \mathbf{A}| \geq k + \delta\sqrt{n}\log n. \tag{2}$$

We assume this is the case for the rest of the proof and fix any such set $A \subset \overline{M}$. It suffices to show that $\boldsymbol{f} = \boldsymbol{f}_{M,A,\mathbf{H}}$ is ϵ-far from k-juntas with probability $1 - o(1)$, where $\mathbf{H}$ is sampled according to the rest (steps 3 and 4) of the procedure for $\mathcal{D}_{\text{no}}$ (by sampling $\mathbf{S}_i$ from A and then $\boldsymbol{h}_i$ over $\mathbf{S}_i$).

The plan for the rest of the proof is the following. For each $V \subset M \cup A$ of size $9\sqrt{n}$, we use $\mathbf{E}_V$ to denote the size of the *maximum* set of vertex-disjoint, $\boldsymbol{f}$-bichromatic edges along directions in V only. We will prove the following claim:

▶ **Claim 5.** *For each $V \subset M \cup A$ of size $9\sqrt{n}$, we have $\mathbf{E}_V \geq \epsilon 2^n$ with probability $1 - \exp(-2^{\Omega(n)})$.*

Note that when $\mathbf{E}_V \geq \epsilon 2^n$, we have $\text{dist}(\boldsymbol{f}, g) \geq \epsilon$ for every function g that does not depend on any variable in V. This is because, for every $\boldsymbol{f}$-bichromatic edge $(x, x^{(\ell)})$ along a coordinate $\ell \in V$, we must have $\boldsymbol{f}(x) \neq \boldsymbol{f}(x^{(\ell)})$ since the edge is bichromatic but $g(x) = g(x^{(\ell)})$ as g does not depend on the ℓth variable. As a result, $\boldsymbol{f}$ must disagree with g on at least $\epsilon 2^n$ many points.

Assuming Claim 5 for now, we can apply a union bound over all

$$\binom{|M \cup A|}{9\sqrt{n}} \leq \binom{n}{9\sqrt{n}} \leq 2^{O(\sqrt{n}\log n)}$$

possible choices of $V \subset M \cup A$ to conclude that with probability $1 - o(1)$, $\boldsymbol{f} = \boldsymbol{f}_{M,A,\mathbf{H}}$ is ϵ-far from all functions that do not depend on at least $9\sqrt{n}$ variables in $M \cup A$. By (2), this set includes all k-juntas. This concludes the proof of the Lemma 4 modulo the proof of Claim 5. ◀

In the rest of the section, we prove Claim 5 for a fixed subset $V \subset M \cup A$ of size $9\sqrt{n}$. We start with the simpler case when $V \cap M$ is nonempty.

▶ **Claim 6.** *If $V \cap M \neq \emptyset$, then we have $\mathbf{E}_V \geq 2^n/5$ with probability $1 - \exp(-2^{\Omega(n)})$.*

Proof. Fix an $\ell \in V \cap M$; we will argue that with probability $1 - \exp(-2^{\Omega(n)})$ there are at least $2^n/5$ $\boldsymbol{f}$-bichromatic edges along direction ℓ. This suffices since such edges are clearly vertex-disjoint.

Observe that since $\ell \in M$, every $x \in \{0,1\}^n$ has $\Gamma(x) \neq \Gamma(x^{(\ell)})$. For each $b \in \{0,1\}^M$, let X_b be the set of $x \in \{0,1\}^n$ with $x_{|S} = b$. We partition $\{0,1\}^n$ into 2^{t-1} pairs X_b and

$X_{b^{(\ell)}}$, where b ranges over the 2^{t-1} strings in $\{0,1\}^M$ with $b_\ell = 0$. For each such pair, we use $\mathbf{D}_b$ to denote the number of $\boldsymbol{f}$-bichromatic edges between X_b and $X_{b^{(\ell)}}$. We are interested in lower bounding $\sum_b \mathbf{D}_b$.

We will apply Hoeffding's inequality. For this purpose we note that the $\mathbf{D}_b$'s are independent (since they depend on distinct $\boldsymbol{h}_i$'s), always lie between 0 and 2^m, and each one has expectation 2^{m-1}. The latter is because each edge $(x, x^{(\ell)})$ has $\boldsymbol{f}(x)$ and $\boldsymbol{f}(x^{(\ell)})$ drawn as two independent random bits, which is the case since $\Gamma(x) \neq \Gamma(x^{(\ell)})$. Thus, the expectation of $\sum_b \mathbf{D}_b$ is 2^{n-2}. By Hoeffding's inequality, we have

$$\Pr\left[\left|\sum \mathbf{D}_b - 2^{n-2}\right| \geq \frac{2^n}{20}\right] \leq 2 \cdot \exp\left(-\frac{2(2^n/20)^2}{2^{t-1} \cdot 2^{2m}}\right) = \exp\left(-2^{\Omega(n)}\right)$$

since $t = \Omega(n)$. This finishes the proof of the claim. ◀

Now we may assume that $V \subset A$ (and $|V| = 9\sqrt{n}$). We use $\mathbf{I}$ to denote the set of $i \in [N]$ such that $\mathbf{S}_i \cap V \neq \emptyset$. The following claim shows that $\mathbf{I}$ is large with extremely high probability:

▶ **Claim 7.** *We have $|\mathbf{I}| \geq 4.4\epsilon N$ with probability at least $1 - \exp(-2^{\Omega(n)})$ over the choice of $\mathbf{S}$.*

Proof. For each $i \in [N]$ we have (using $1 - x \leq e^{-x}$ for all x and $1 - x/2 \geq e^{-x}$ for $x \in [0, 1.5]$):

$$\mathbf{Pr}\left[i \in \mathbf{I}\right] = 1 - \left(1 - \frac{\epsilon}{\sqrt{n}}\right)^{9\sqrt{n}} \geq 1 - e^{-9\epsilon} \geq 4.5\epsilon,$$

since $\epsilon/\sqrt{n}$ is the probability of each element of A being included in $\mathbf{S}_i$ and $\epsilon \leq 1/6$ so $9\epsilon \leq 1.5$.

Using $\epsilon \geq 2^{-(2\alpha-1)n/2}$ from (1), we have $\mathbf{E}[|\mathbf{I}|] \geq 4.5\epsilon N = 2^{\Omega(n)}$. Since the $\mathbf{S}_i$'s are independent, a Chernoff bound implies that $|\mathbf{I}| \geq 4.4\epsilon N$ with probability $1 - \exp(-2^{\Omega(n)})$. ◀

By Claim 7, we fix $S_1, \ldots, S_N$ to be any sequence of subsets of A that satisfy $|I| \geq 4.4\epsilon N$ in the rest of the proof, and it suffices to show that over the random choices of $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N$ (where each $\boldsymbol{h}_i$ is chosen to be a random function over S_i), $\mathbf{E}_V \geq \epsilon 2^n$ with probability at least $1 - \exp(-2^{\Omega(n)})$.

To this end we use $\rho(i)$ for each $i \in I$ to denote the first coordinate of S_i in V, and Z_i to denote the set of $x \in \{0,1\}^n$ with $\Gamma(x) = i$. Note that the Z_i's are disjoint. We further partition each Z_i into disjoint $Z_{i,b}$, $b \in \{0,1\}^{S_i}$, with $x \in Z_{i,b}$ iff $x \in Z_i$ and $x_{|S_i} = b$. For each $i \in I$ and $b \in \{0,1\}^{S_i}$ with $b_{\rho(i)} = 0$, we use $\mathbf{D}_{i,b}$ to denote the number of $\boldsymbol{f}$-bichromatic edges between $Z_{i,b}$ and $Z_{i,b^{(\rho(i))}}$ along the $\rho(i)$th direction. It is clear that such edges, over all i and b, are vertex-disjoint and thus,

$$\mathbf{E}_V \geq \sum_{i \in I} \sum_{\substack{b \in \{0,1\}^{S_i} \\ b_{\rho(i)} = 0}} \mathbf{D}_{i,b}. \tag{3}$$

We will apply Hoeffding's inequality. Note that $\mathbf{D}_{i,b}$ is $2^{m-|S_i|}$ with probability $1/2$, and 0 with probability $1/2$. Thus, the expectation of the RHS of (3) is

$$\sum_{i \in I} 2^{|S_i|-1} \cdot 2^{m-|S_i|-1} = |I| \cdot 2^{m-2} \geq 1.1\epsilon 2^n,$$

using $|I| \geq 4.4\epsilon N$. Since all the $\mathbf{D}_{i,b}$'s are independent, by Hoeffding's inequality we have

$$\Pr\left[\left|\text{RHS of (3)} - |I|\cdot 2^{m-2}\right| \geq 0.01|I|\cdot 2^{m-2}\right] \leq 2\cdot\exp\left(-\frac{2(0.01|I|\cdot 2^{m-2})^2}{\sum_{i\in I} 2^{|S_i|-1}\cdot 2^{2(m-|S_i|)}}\right) \leq \exp\left(-2^{\Omega(n)}\right),$$

since $|I| \geq \Omega(\epsilon N) = 2^{\Omega(n)}$. When this does not happen, we have $\mathbf{E}_V \geq 0.99\cdot|I|\cdot 2^{m-2} > \epsilon 2^n$.

3 The Set-Size-Set-Queries (SSSQ) Problem

We first introduce the Set-Size-Set-Queries (SSSQ for short) problem, which is an artificial problem that we use as a bridge to prove Theorem 2. We use the same parameters p, q and m from the definition of $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$, with n being sufficiently large (so $m = \Omega(n)$ is sufficiently large as well).

We start by defining $\mathcal{A}_{\text{yes}}$ and $\mathcal{A}_{\text{no}}$, two distributions over subsets of $[m]$: $\mathbf{A} \sim \mathcal{A}_{\text{yes}}$ is drawn by independently including each element of $[m]$ with probability p and $\mathbf{A} \sim \mathcal{A}_{\text{no}}$ is drawn by independently including each element with probability q. In SSSQ, the algorithm needs to determine whether an unknown $A \subseteq [m]$ is drawn from $\mathcal{A}_{\text{yes}}$ or $\mathcal{A}_{\text{no}}$. (For intuition, to see that this task is reasonable, we observe here that a straightforward Chernoff bound shows that almost every outcome of $\mathbf{A} \sim \mathcal{A}_{\text{yes}}$ is larger than almost every outcome of $\mathbf{A} \sim \mathcal{A}_{\text{no}}$ by $\Omega(\sqrt{n}\log n)$.)

Let A be a subset of $[m]$ which is hidden in an oracle. An algorithm accesses A (in order to tell whether it is drawn from $\mathcal{A}_{\text{yes}}$ or $\mathcal{A}_{\text{no}}$) by interacting with the oracle in the following way: each time it calls the oracle, it does so by sending a subset of $[m]$ to the oracle. The oracle responds as follows: for each j in the subset, it returns a bit that is 0 if $j \notin A$, and is 1 with probability $\epsilon/\sqrt{n}$ and 0 with probability $1 - \epsilon/\sqrt{n}$ if $j \in A$. The cost of such an oracle call is the size of the subset provided to the oracle.

More formally, a deterministic and non-adaptive algorithm $\textsc{Alg} = (g, T)$ for SSSQ accesses the set A hidden in the oracle by submitting a list of queries $T = (T_1, \ldots, T_d)$, for some $d \geq 1$, where each $T_i \subseteq [m]$ is a set. (Thus, we call each T_i a *set query*, as part of the name SSSQ.)

- Given T, the oracle returns a list of random vectors $\boldsymbol{v} = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_d)$, where $\boldsymbol{v}_i \in \{0,1\}^{T_i}$ and each bit $\boldsymbol{v}_{i,j}$ is independently distributed as follows: if $j \notin A$ then $\boldsymbol{v}_{i,j} = 0$, and if $j \in A$ then

$$\boldsymbol{v}_{i,j} = \begin{cases} 1 & \text{with probability } \epsilon/\sqrt{n} \\ 0 & \text{with probability } 1 - (\epsilon/\sqrt{n}). \end{cases} \tag{4}$$

  Note that the random vectors in $\boldsymbol{v}$ depend on both T and A.
- Given $\boldsymbol{v} = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_d)$, $\textsc{Alg}$ returns (deterministically) the value of $g(\boldsymbol{v}) \in \{\text{“yes”}, \text{“no”}\}$.

The performance of $\textsc{Alg} = (g, T)$ is measured by its *query complexity* and its *advantage*.

- The query complexity of $\textsc{Alg}$ is defined as $\sum_{i=1}^{d} |T_i|$, the total size of all the set queries. On the other hand, the advantage of $\textsc{Alg}$ is defined as

$$\Pr_{\mathbf{A}\sim\mathcal{A}_{\text{yes}}}\left[\textsc{Alg}(\mathbf{A}) = \text{“yes”}\right] - \Pr_{\mathbf{A}\sim\mathcal{A}_{\text{no}}}\left[\textsc{Alg}(\mathbf{A}) = \text{“yes”}\right].$$

▶ Remark 8. In the definition above, g is a deterministic map from all possible sequences of vectors returned by the oracle to "yes" or "no." Considering only deterministic as opposed to randomized g is without loss of generality since given any query sequence T, the highest possible advantage can always be achieved by a deterministic map g.

We prove the following lower bound for any deterministic, non-adaptive ALG in Section 5.

▶ **Lemma 9.** *Any deterministic, non-adaptive* ALG *for* SSSQ *with advantage at least* $2/3$ *satisfies*

$$\sum_{i=1}^{d} |T_i| \geq \frac{n^{3/2}}{\epsilon \cdot \log^3 n \cdot \log^2(n/\epsilon)}.$$

4 Reducing from SSSQ to distinguishing $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$

In this section we reduce from SSSQ to the problem of distinguishing the pair of distributions $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$. More precisely, let $\text{ALG}^* = (h, X)$ denote a deterministic and nonadaptive algorithm that makes $q \leq (n/\epsilon)^2$ string queries[1] $X = (x_1, \ldots, x_q)$ to a hidden function f drawn from either $\mathcal{D}_{\text{yes}}$ or $\mathcal{D}_{\text{no}}$, applies the (deterministic) map h to return $h(f(x_1), \ldots, f(x_q)) \in \{\text{"yes"}, \text{"no"}\}$, and satisfies

$$\Pr_{\boldsymbol{f} \sim \mathcal{D}_{\text{yes}}} \left[\text{ALG}^*(\boldsymbol{f}) = \text{"yes"}\right] - \Pr_{\boldsymbol{f} \sim \mathcal{D}_{\text{no}}} \left[\text{ALG}^*(\boldsymbol{f}) = \text{"yes"}\right] \geq 3/4. \tag{5}$$

We show how to define from $\text{ALG}^* = (h, X)$ an algorithm $\text{ALG} = (g, T)$ for the problem SSSQ with query complexity at most $\tau \cdot q$ and advantage $2/3$, where $\tau = c_\alpha \cdot 5 \log(n/\epsilon)$ and

$$c_\alpha = -\frac{1}{\log(1.5 - \alpha)} > 0 \quad \text{with} \quad (1.5 - \alpha)^{c_\alpha} = 1/2$$

is a constant that depends on α. Given this reduction it follows from Lemma 9 that $q \geq \widetilde{\Omega}(n^{3/2}/\epsilon)$. This finishes the proof of Theorem 2.

We start with some notation. Recall that in both $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$, $\mathbf{M}$ is a subset of $[n]$ of size t drawn uniformly at random. For a fixed M of size t, we use $\mathcal{E}_{\text{yes}}(M)$ to denote the distribution of $\mathbf{A}$ and $\mathbf{H}$ sampled in the randomized procedure for $\mathcal{D}_{\text{yes}}$, conditioning on $\mathbf{M} = M$. We define $\mathcal{E}_{\text{no}}(M)$ similarly. Then conditioning on $\mathbf{M} = M$, $\boldsymbol{f} \sim \mathcal{D}_{\text{yes}}$ is distributed as $f_{M,\mathbf{A},\mathbf{H}}$ with $(\mathbf{A}, \mathbf{H}) \sim \mathcal{E}_{\text{yes}}(M)$ and $\boldsymbol{f} \sim \mathcal{D}_{\text{no}}$ is distributed as $f_{M,\mathbf{A},\mathbf{H}}$ with $(\mathbf{A}, \mathbf{H}) \sim \mathcal{E}_{\text{no}}(M)$. This allows us to rewrite (5) as

$$\frac{1}{\binom{n}{t}} \cdot \sum_{M:|M|=t} \left(\Pr_{(\mathbf{A},\mathbf{H}) \sim \mathcal{E}_{\text{yes}}(M)} \left[\text{ALG}^*(\boldsymbol{f}_{M,\mathbf{A},\mathbf{H}}) = \text{"yes"}\right] - \Pr_{(\mathbf{A},\mathbf{H}) \sim \mathcal{E}_{\text{no}}(M)} \left[\text{ALG}^*(\boldsymbol{f}_{M,\mathbf{A},\mathbf{H}}) = \text{"yes"}\right] \right) \geq \frac{3}{4}.$$

We say $M \subset [n]$ is *good* if any two queries x_i and x_j in X with Hamming distance $\|x_i - x_j\|_1 \geq \tau$ have different projections on M, i.e., $(x_i)_{|M} \neq (x_j)_{|M}$. We prove below that most M's are good.

▶ **Claim 10.** $\Pr_{\mathbf{M}} \left[\mathbf{M} \textit{ is not good}\right] = o(1)$.

[1] Any algorithm that makes more than this many queries already fits the $\widetilde{\Omega}(n^{3/2}/\epsilon)$ lower bound we aim for.

Proof. For each pair of strings x_i and x_j in X with Hamming distance at least τ, the probability of them having the same projection on $\mathbf{M}$ (drawn uniformly from all size-t subsets) is at most

$$\frac{\binom{n-\tau}{t}}{\binom{n}{t}} = \frac{(n-\tau-t+1)\cdots(n-t)}{(n-\tau+1)\cdots n} \leq \left(1-\frac{t}{n}\right)^{\tau} \leq \big(2(1-\alpha)+o(1)\big)^{\tau} < (1.5-\alpha)^{\tau} \leq O\left(\frac{\epsilon}{n}\right)^{5},$$

by our choices of c_α and τ. The claim follows by a union bound over at most $q^2 \leq (n/\epsilon)^4$ pairs. ◀

We can split the sum (5) into two sums: the sum over good M and the sum over bad M. By Claim 10 the contribution from the bad M is at most $o(1)$, and thus we have that

$$\frac{1}{\binom{n}{t}} \cdot \sum_{\text{good } M} \left(\mathop{\mathbf{Pr}}_{(\mathbf{A},\mathbf{H})\sim\mathcal{E}_{\text{yes}}(M)} \left[\textsc{Alg}^*(\boldsymbol{f}_{M,\mathbf{A},\mathbf{H}}) = \text{“yes”}\right] - \mathop{\mathbf{Pr}}_{(\mathbf{A},\mathbf{H})\sim\mathcal{E}_{\text{no}}(M)} \left[\textsc{Alg}^*(\boldsymbol{f}_{M,\mathbf{A},\mathbf{H}}) = \text{“yes”}\right] \right)$$

is at least $3/4 - o(1)$. Thus, there must exist a good set $M \subset [n]$ of size t with

$$\mathop{\mathbf{Pr}}_{(\mathbf{A},\mathbf{H})\sim\mathcal{E}_{\text{yes}}(M)} \left[\textsc{Alg}^*(\boldsymbol{f}_{M,\mathbf{A},\mathbf{H}}) = \text{“yes”}\right] - \mathop{\mathbf{Pr}}_{(\mathbf{A},\mathbf{H})\sim\mathcal{E}_{\text{no}}(M)} \left[\textsc{Alg}^*(\boldsymbol{f}_{M,\mathbf{A},\mathbf{H}}) = \text{“yes”}\right] \geq 2/3. \tag{6}$$

Fix such a good M. We use $\textsc{Alg}^* = (h, X)$ and M to define an algorithm $\textsc{Alg} = (g, T)$ for SSSQ as follows (note that the algorithm $\textsc{Alg}$ below actually works over the universe $\overline{M}$ (of size m) instead of $[m]$ as in the original definition of SSSQ but this can be handled by picking any bijection between $\overline{M}$ and $[m]$; accordingly $\mathbf{A} \sim \mathcal{A}_{\text{yes}}$ is drawn by including each element of $\overline{M}$ with probability p and $\mathbf{A} \sim \mathcal{A}_{\text{no}}$ is drawn by including each element of $\overline{M}$ with probability q). We start with T:

1. First we use M to define an equivalence relation $\sim$ over the query set X, where $x_i \sim x_j$ if $(x_i)_{|M} = (x_j)_{|M}$. Let $X_1, \ldots, X_d$, $d \geq 1$, denote the equivalence classes of X, and let us write $\rho(\ell)$ for each $\ell \in [d]$ to denote the value $\Gamma(x) \in [N]$ that is shared by all strings $x \in X_\ell$.
2. Next we define a sequence of subsets of $\overline{M}$, $T = (T_1, \ldots, T_d)$, as the set queries of $\textsc{Alg}$, where

$$T_\ell = \big\{ i \in \overline{M} \colon \exists\, x, y \in X_\ell \text{ such that } x_i \neq y_i \big\}. \tag{7}$$

To upper bound $|T_\ell|$, fixing an arbitrary string $x \in X_\ell$ and recalling that M is good, we have that

$$|T_\ell| \leq \sum_{y \in X_\ell} \|x-y\|_1 \leq \sum_{y \in X_\ell} \tau = \tau \cdot |X_\ell|.$$

As a result, the query complexity of $\textsc{Alg}$ (using T as its set queries) is at most

$$\sum_{\ell=1}^{d} |T_\ell| \leq \tau \cdot \sum_{\ell=1}^{d} |X_\ell| \leq \tau \cdot q.$$

It remains to define h and then prove that the advantage of $\textsc{Alg} = (g, T)$ for SSSQ is at least $2/3$. Indeed the g that we define is a randomized map and we describe it as a randomized procedure below (by Remark 8 one can extract from g a deterministic map that achieves the same advantage):

1. Given $v_1, \ldots, v_d$, $v_\ell \in \{0,1\}^{T_\ell}$, as the strings returned by the oracle upon being given T, let

$$R_\ell = \{j \in T_\ell : v_{\ell,j} = 1\}. \tag{8}$$

   For each $\ell \in [d]$, the procedure draws a random function $\boldsymbol{f}_\ell : \{0,1\}^{R_\ell} \to \{0,1\}$, by flipping $2^{|R_\ell|}$ many independent and unbiased random bits.
2. Next for each query $x \in X_\ell$, $\ell \in [d]$, we feed $\boldsymbol{f}_\ell(x_{|R_\ell})$ to h as the bit that the oracle returns upon the query x. Finally the procedure returns the result ("yes" or "no") that h returns.

In the rest of the proof we show that the advantage of $\textsc{Alg} = (g, T)$ is exactly the same as the LHS of (6) and thus, is at least 2/3.

For convenience, we use $\mathcal{V}_{\text{yes}}$ to denote the distribution of responses $\boldsymbol{v} = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_d)$ to T when $\mathbf{A} \sim \mathcal{A}_{\text{yes}}$, and $\mathcal{V}_{\text{no}}$ to denote the distribution when $\mathbf{A} \sim \mathcal{A}_{\text{no}}$. Then the advantage of $\textsc{Alg}$ is

$$\Pr_{\boldsymbol{v} \sim \mathcal{V}_{\text{yes}}}\left[g(\boldsymbol{v}) = \text{“yes”}\right] - \Pr_{\boldsymbol{v} \sim \mathcal{V}_{\text{no}}}\left[g(\boldsymbol{v}) = \text{“yes”}\right].$$

It suffices to show that

$$\Pr_{\boldsymbol{v} \sim \mathcal{V}_{\text{yes}}}\left[g(\boldsymbol{v}) = \text{“yes”}\right] = \Pr_{(\mathbf{A},\mathbf{H}) \sim \mathcal{E}_{\text{yes}}(M)}\left[\textsc{Alg}^*(\boldsymbol{f}_{M,\mathbf{A},\mathbf{H}}) = \text{“yes”}\right] \quad \text{and} \tag{9}$$

$$\Pr_{\boldsymbol{v} \sim \mathcal{V}_{\text{no}}}\left[g(\boldsymbol{v}) = \text{“yes”}\right] = \Pr_{(\mathbf{A},\mathbf{H}) \sim \mathcal{E}_{\text{no}}(M)}\left[\textsc{Alg}^*(\boldsymbol{f}_{M,\mathbf{A},\mathbf{H}}) = \text{“yes”}\right]. \tag{10}$$

We show (9); the proof of (10) is similar. From the definition of $\mathcal{V}_{\text{yes}}$ and $\mathcal{E}_{\text{yes}}(M)$ the distribution of $(\mathbf{R}_\ell : \ell \in [d])$ derived from $\boldsymbol{v} \sim \mathcal{V}_{\text{yes}}$ using (8) is the same as the distribution of $(\mathbf{S}_{\rho(\ell)} \cap T_\ell : \ell \in [d])$: both are sampled by first drawing a random subset $\mathbf{A}$ of $\overline{M}$ and then drawing a random subset of $\mathbf{A} \cap T_\ell$ independently by including each element of $\mathbf{A} \cap T_\ell$ with the same probability $\epsilon/\sqrt{n}$ (recall in particular equation (4) and step 3 of the randomized procedure specifying $\mathcal{D}_{\text{yes}}$ in Section 2). Since $\boldsymbol{f}_{M,\mathbf{A},\mathbf{H}}(x)$ for $x \in X_\ell$ is determined by a random Boolean function $\boldsymbol{h}_{\rho(\ell)}$ from $\{0,1\}^{\mathbf{S}_{\rho(\ell)}}$ to $\{0,1\}$, and since all the queries in X_ℓ only differ by coordinates in T_ℓ, the distribution of the q bits that g feeds to h when $\boldsymbol{v} \sim \mathcal{V}_{\text{yes}}$ is the same as the distribution of $(\boldsymbol{f}(x) : x \in X)$ when $\boldsymbol{f} \sim \mathcal{E}_{\text{yes}}(M)$. This finishes the proof of (9), and concludes our reduction argument.

5 A lower bound on the non-adaptive query complexity of SSSQ

We will prove Lemma 9 by first giving a reduction from an even simpler algorithmic task, which we describe next in Section 5.1. We will then prove a lower bound for the simpler task in Section 5.2.

5.1 Set-Size-Element-Queries (SSEQ)

Recall the parameters m, p, q and ϵ and the two distributions $\mathcal{A}_{\text{yes}}$ and $\mathcal{A}_{\text{no}}$ used in the definition of problem SSSQ. We now introduce a simpler algorithmic task called the Set-Size-Element-Queries (SSEQ) problem using the same parameters and distributions.

Let A be a subset of $[m]$ hidden in an oracle. An algorithm accesses the oracle to tell whether it is drawn from $\mathcal{A}_{\text{yes}}$ or $\mathcal{A}_{\text{no}}$. The difference between SSSQ and SSEQ is the way an algorithm accesses A. In SSEQ, an algorithm $\textsc{Alg}' = (h, \ell)$ submits a vector $\ell = (\ell_1, \ldots, \ell_m)$ of nonnegative integers.

- On receiving ℓ, the oracle returns a random response vector $\boldsymbol{b} \in \{0,1\}^m$, where each entry $\boldsymbol{b}_i$ is distributed independently as follows: if $i \notin A$ then $\boldsymbol{b}_i = 0$, and if $i \in A$ then

$$\boldsymbol{b}_i = \begin{cases} 1 & \text{with probability } \lambda(\ell_i) \\ 0 & \text{with probability } 1-\lambda(\ell_i) \end{cases}, \qquad \text{where } \lambda(\ell_i) = 1 - \left(1 - \frac{\epsilon}{\sqrt{n}}\right)^{\ell_i}.$$

  Equivalently, for each $i \in A$, the oracle independently flips ℓ_i coins, each of which is 1 with probability $\epsilon/\sqrt{n}$, and at the end returns $\boldsymbol{b}_i = 1$ to the algorithm if and only if at least one of the coins is 1. Thus, we refer to each ℓ_i as ℓ_i *element-queries* for the ith element.
- After receiving the vector $\boldsymbol{b}$ from the oracle, $\textsc{Alg}'$ returns the value $h(\boldsymbol{b}) \in \{\text{“yes”}, \text{“no”}\}$. Here h is a deterministic map from $\{0,1\}^m$ to $\{\text{“yes”}, \text{“no”}\}$.

Similar to before, the performance of $\textsc{Alg}'$ is measured by its query complexity and its advantage:

- The query complexity of $\textsc{Alg}' = (h, \ell)$ is defined as $\|\ell\|_1 = \sum_{i=1}^m \ell_i$. For its advantage, we let $\mathcal{B}_{\text{yes}}$ denote the distribution of response vectors $\boldsymbol{b}$ to query ℓ when $\mathbf{A} \sim \mathcal{A}_{\text{yes}}$, and $\mathcal{B}_{\text{no}}$ denote the distribution when $\mathbf{A} \sim \mathcal{D}_{\text{no}}$. The advantage of $\textsc{Alg}' = (h, \ell)$ is then defined as

$$\Pr_{\boldsymbol{b} \sim \mathcal{B}_{\text{yes}}}\left[h(\boldsymbol{b}) = \text{“yes”}\right] - \Pr_{\boldsymbol{b} \sim \mathcal{B}_{\text{no}}}\left[h(\boldsymbol{b}) = \text{“yes”}\right].$$

▶ Remark 11. It is worth pointing out (we will use it later) that the highest possible advantage over all deterministic maps h is a monotonically non-decreasing function of the coordinates of ℓ. To see this, let A be the underlying set and let ℓ and ℓ' be two vectors with $\ell_i \le \ell'_i$ for every $i \in [m]$. Let $\boldsymbol{b}$ and $\boldsymbol{b}'$ be the random vectors returned by the oracle upon ℓ and ℓ'. Then we can define $\boldsymbol{b}^*$ using $\boldsymbol{b}'$ as follows: $\boldsymbol{b}^*_i = 0$ if $\boldsymbol{b}'_i = 0$; otherwise when $\boldsymbol{b}'_i = 1$, we set

$$\boldsymbol{b}^*_i = \begin{cases} 1 & \text{with probability } \lambda(\ell_i)/\lambda(\ell'_i) \\ 0 & \text{with probability } 1-\lambda(\ell_i)/\lambda(\ell'_i) \end{cases}.$$

One can verify that the distribution of $\boldsymbol{b}$ is exactly the same as the distribution of $\boldsymbol{b}^*$. Hence there is a randomized map h' such that the advantage of (h', ℓ') is at least as large as the highest possible advantage achievable using ℓ. The remark now follows by our earlier observation in Remark 8 that the highest possible advantage using ℓ' is always achieved by a deterministic h'.

The following lemma reduces the proof of Lemma 9 to proving a lower bound for SSEQ.

▶ **Lemma 12.** *Given any deterministic and non-adaptive algorithm* $\textsc{Alg} = (g, T)$ *for* SSSQ*, there is a deterministic and non-adaptive algorithm* $\textsc{Alg}' = (h, \ell)$ *for* SSEQ *with the same query complexity as* Alg *and advantage at least as large as that of* Alg*.*

Proof. We show how to construct $\textsc{Alg}' = (h, \ell)$ from $\textsc{Alg} = (g, T)$, where h is a randomized map, such that $\textsc{Alg}'$ has exactly the same query complexity and advantage as those of Alg. The lemma then follows from the observation we made earlier in Remark 8.

We define ℓ first. Given $T = (T_1, \ldots, T_d)$ for some $d \ge 1$, $\ell = (\ell_1, \ldots, \ell_m)$ is defined as

$$\ell_j = \left|\{i \in [d] : j \in T_i\}\right|.$$

So $\|\ell\|_1 = \sum_{i=1}^d |T_i|$. To define h we describe a randomized procedure P that, given any $b \in \{0,1\}^m$, outputs a sequence of random vectors $\boldsymbol{v} = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_d)$ such that the following claim holds.

▶ **Claim 13.** *If $\boldsymbol{b} \sim \mathcal{B}_{yes}$ (or $\mathcal{B}_{no}$), then $P(\boldsymbol{b})$ is distributed the same as $\mathcal{V}_{yes}$ (or $\mathcal{V}_{no}$, respectively).*

Assuming Claim 13, we can set $h = g \circ P$ and the advantage of $\textsc{Alg}'$ would be the same as that of $\textsc{Alg}$. In the rest of the proof, we describe the randomized procedure P and prove Claim 13.

Given $b \in \{0,1\}^m$, P outputs a sequence of random vectors $\boldsymbol{v} = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_d)$ as follows:

- If $b_j = 0$, then for each $i \in [d]$ with $j \in T_i$, P sets $\boldsymbol{v}_{i,j} = 0$.
- If $b_j = 1$ (this implies that $\ell_j > 0$ and $j \in T_i$ for some $i \in [d]$), P sets $(\boldsymbol{v}_{i,j} : i \in [d], j \in T_i)$ to be a length-r, where $r = |\{i \in [d] : j \in T_i\}|$, binary string in which each bit is independently 1 with probability $\epsilon/\sqrt{n}$ and 0 with probability $1 - \epsilon/\sqrt{n}$, conditioned on its not being 0^r.

Proof of Claim 13. It suffices to prove that, fixing any $A \subseteq [m]$ as the underlying set hidden in the oracle, the distribution of $\boldsymbol{v}$ is the same as the distribution of $P(\boldsymbol{b})$. The claim then follows since in the definitions of both $\mathcal{B}_{\text{yes}}$ and $\mathcal{V}_{\text{yes}}$ (or $\mathcal{B}_{\text{no}}$ and $\mathcal{V}_{\text{no}}$), A is drawn from $\mathcal{A}_{\text{yes}}$ (or $\mathcal{A}_{\text{no}}$, respectively).

Consider a sequence v of d vectors $v_1, \ldots, v_d$ with $v_i \in \{0,1\}^{T_i}$ for each $i \in [d]$, and let

$$n_{j,1} = |\{i \in [d] : j \in T_i \text{ and } v_{i,j} = 1\}| \quad \text{and} \quad n_{j,0} = |\{i \in [d] : j \in T_i \text{ and } v_{i,j} = 0\}|,$$

for each $j \in [m]$. Then the $\boldsymbol{v}$ returned by the oracle (in SSSQ) is equal to v with probability:

$$\mathbf{1}\{\forall j \notin A, n_{j,1} = 0\} \cdot \prod_{j \in A} \left(\frac{\epsilon}{\sqrt{n}}\right)^{n_{j,1}} \left(1 - \frac{\epsilon}{\sqrt{n}}\right)^{n_{j,0}}, \tag{11}$$

since all coordinates $\boldsymbol{v}_{i,j}$ are independent. On the other hand, the probability of $P(\boldsymbol{b}) = v$ is

$$\mathbf{1}\{\forall j \notin A, n_{j,1} = 0\} \cdot \prod_{j \in A} \left(\mathbf{1}\{n_{j,0} = \ell_j\} \cdot \left(1 - \frac{\epsilon}{\sqrt{n}}\right)^{\ell_j} + \mathbf{1}\{n_{j,1} \geq 1\} \cdot \left(\frac{\epsilon}{\sqrt{n}}\right)^{n_{j,1}} \left(1 - \frac{\epsilon}{\sqrt{n}}\right)^{n_{j,0}} \right),$$

which is exactly the same as the probability of $\boldsymbol{v} = v$ in (11). ◀

This finishes the proof of Lemma 12. ◀

5.2 A lower bound for SSEQ

We prove the following lower bound for SSEQ, from which Lemma 9 follows:

▶ **Lemma 14.** *Any deterministic, non-adaptive $\textsc{Alg}'$ for SSEQ with advantage at least $2/3$ satisfies*

$$\|\ell\|_1 > s \stackrel{\text{def}}{=} \frac{n^{3/2}}{\epsilon \cdot \log^3 n \cdot \log^2(n/\epsilon)}.$$

Proof. Assume for contradiction that there is an algorithm $\textsc{Alg}' = (h, \ell)$ with $\|\ell\|_1 \leq s$ and advantage at least $2/3$. Let ℓ^* be the vector obtained from ℓ by rounding each positive ℓ_i to the smallest power of 2 that is at least as large as ℓ_i (and taking $\ell^*_i = 0$ if $\ell_i = 0$). From Remark 11, there must be a map h^* such that (h^*, ℓ^*) also has advantage at least $2/3$ but now we have 1) $\|\ell^*\|_1 \leq 2s$ and 2) every positive entry of ℓ^* is a power of 2. Below we abuse notation and still use $\textsc{Alg}' = (h, \ell)$ to denote (h^*, ℓ^*): $\textsc{Alg}' = (h, \ell)$ satisfies $\|\ell\|_1 \leq 2s$, every

positive entry of ℓ is a power of 2, and has advantage at least $2/3$. We obtain a contradiction below by showing that any such ℓ can only have an advantage of $o(1)$.

Let $L = \lceil \log(2s) \rceil = O(\log(n/\epsilon))$. Given that $\|\ell\|_1 \le 2s$ we can partition $\{i \in [m] : \ell_i > 0\}$ into $L+1$ sets $C_0, \ldots, C_L$, where bin C_j contains those coordinates $i \in [m]$ with $\ell_i = 2^j$. We may make two further assumptions on $\text{ALG}' = (h, \ell)$ that will simplify the lower bound proof:

- We may reorder the entries in decreasing order and assume without loss of generality that

$$\ell = \left(\underbrace{2^L, \ldots, 2^L}_{c_L}, \underbrace{2^{L-1}, \ldots, 2^{L-1}}_{c_{L-1}}, \ldots, \underbrace{1, \ldots, 1}_{c_0}, 0, \ldots, 0 \right), \tag{12}$$

  where $c_j = |C_j|$ satisfies $\sum_j c_j \cdot 2^j \le 2s$. This is without loss of generality since $\mathcal{A}_{\text{yes}}$ and $\mathcal{A}_{\text{no}}$ are symmetric in the coordinates (and so are $\mathcal{B}_{\text{yes}}$ and $\mathcal{B}_{\text{no}}$).
- For the same reason we may assume that the map $h(b)$ depends only on the number of 1's of b in each set C_j, which we refer to as the *summary* $S(b)$ of b:

$$S(b) \stackrel{\text{def}}{=} \left(\|b_{|C_L}\|_1, \|b_{|C_{L-1}}\|_1, \ldots, \|b_{|C_0}\|_1 \right) \in \mathbb{Z}_{\ge 0}^{L+1}.$$

  To see that this is without loss of generality, consider a randomized procedure P that, given $b \in \{0,1\}^m$, applies an independent random permutation over the entries of C_j for each bin $j \in [0 : L]$. One can verify that the random map $h' = h \circ P$ only depends on the summary $S(b)$ of b but achieves the same advantage as h.

Given a query ℓ as in (12), we define $\mathcal{S}_{\text{yes}}$ to be the distribution of $S(\boldsymbol{b})$ for $\boldsymbol{b} \sim \mathcal{B}_{\text{yes}}$ (recall that $\mathcal{B}_{\text{yes}}$ is the distribution of the vector $\boldsymbol{b}$ returned by the oracle upon the query ℓ when $\mathbf{A} \sim \mathcal{A}_{\text{yes}}$). Similarly we define $\mathcal{S}_{\text{no}}$ as the distribution of $S(\boldsymbol{b})$ for $\boldsymbol{b} \sim \mathcal{B}_{\text{no}}$. As h only depends on the summary the advantage is at most $d_{\text{TV}}(\mathcal{S}_{\text{yes}}, \mathcal{S}_{\text{no}})$, which we upper bound below by $o(1)$.

From the definition of $\mathcal{B}_{\text{yes}}$ (or $\mathcal{B}_{\text{no}}$, respectively) and the fact that $\mathcal{A}_{\text{yes}}$ (or $\mathcal{A}_{\text{no}}$, respectively) is symmetric over the m coordinates, we have that the $L+1$ entries of $\mathcal{S}_{\text{yes}}$ (of $\mathcal{S}_{\text{no}}$, respectively) are mutually independent, and that their entries for each C_j, $j \in [0 : L]$, are distributed as $\text{Bin}(c_j, p\lambda_j)$ (as $\text{Bin}(c_j, q\lambda_j)$, respectively), where we have $\lambda_j = 1 - (1 - (\epsilon/\sqrt{n}))^{2^j}$.

In order to prove that $d_{\text{TV}}(\mathcal{S}_{\text{yes}}, \mathcal{S}_{\text{no}}) = o(1)$ and achieve the desired contradiction, we will give upper bounds on the total variation distance between their C_j-entries for each $j \in \{0, \ldots, L\}$.

▶ **Claim 15.** *Let* $\mathbf{X} \sim \text{Bin}(c_j, p\lambda_j)$ *and* $\mathbf{Y} \sim \text{Bin}(c_j, q\lambda_j)$. *Then* $d_{\text{TV}}(\mathbf{X}, \mathbf{Y}) \le o\left(1/L\right)$.

We delay the proof of Claim 15, but assuming it we may simply apply the following well-known proposition to conclude that $d_{\text{TV}}(\mathcal{S}_{\text{yes}}, \mathcal{S}_{\text{no}}) = o(1)$.

▶ **Proposition 16** (Subadditivity of total variation distance)**.** *Let* $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_k)$ *and* $\mathbf{Y} = (\mathbf{Y}_1, \ldots, \mathbf{Y}_k)$ *be two tuples of independent random variables. Then* $d_{\text{TV}}(\mathbf{X}, \mathbf{Y}) \le \sum_{i=1}^k d_{\text{TV}}(\mathbf{X}_i, \mathbf{Y}_i)$.

This gives us a contradiction and finishes the proof of Lemma 14. ◀

Below we prove Claim 15.

Proof of Claim 15. The claim is trivial when $c_j = 0$ so we assume below that $c_j > 0$.

Let $r = p\lambda_j$ and $x = \log n \cdot \lambda_j/\sqrt{n}$. Then $\mathbf{X} \sim \mathrm{Bin}(c_j, r)$ and $\mathbf{Y} \sim \mathrm{Bin}(c_j, r + x)$. As indicated in Equation (2.15) of [1], Equation (15) of [37] gives

$$d_{\mathrm{TV}}(\mathbf{X}, \mathbf{Y}) \leq O\left(\frac{\tau(x)}{(1-\tau(x))^2}\right), \quad \text{where} \quad \tau(x) \stackrel{\text{def}}{=} x\sqrt{\frac{c_j + 2}{2r(1-r)}}, \tag{13}$$

whenever $\tau(x) < 1$. Substituting for x and r, we have (using $c_j \geq 1$, $r \leq 1/2$ and $p = 1/2$)

$$\tau(x) = O\left(\frac{\log n \cdot \lambda_j}{\sqrt{n}} \cdot \sqrt{\frac{c_j}{r}}\right) = O\left(\log n \cdot \sqrt{\frac{\lambda_j \cdot c_j}{n}}\right) = O\left(\frac{1}{L} \cdot \sqrt{\frac{n^{1/2} \cdot \lambda_j}{2^j \cdot \epsilon \cdot \log n}}\right),$$

where the last inequality follows from

$$c_j \cdot 2^j \leq 2s \leq O\left(\frac{n^{3/2}}{\epsilon \cdot \log^3 n \cdot L^2}\right).$$

Finally, note that (using $1 - x > e^{-2x}$ for small positive x and $1 - x \leq e^{-x}$ for all x):

$$1 - \lambda_j = \left(1 - \frac{\epsilon}{\sqrt{n}}\right)^{2^j} \geq \left(e^{-2\epsilon/\sqrt{n}}\right)^{2^j} = e^{-2^{j+1}\epsilon/\sqrt{n}} \geq 1 - O(2^j\epsilon/\sqrt{n})$$

and $\dfrac{\sqrt{n} \cdot \lambda_j}{2^j \cdot \epsilon} = O(1)$. This implies $\tau(x) = o(1/L) = o(1)$. The claim then follows from (13). ◀

References

1 José A Adell and Pedro Jodrá. Exact kolmogorov and total variation distances between some familiar discrete distributions. *Journal of Inequalities and Applications*, 2006(1):1–8, 2006.

2 N. Alon, T. Kaufman, M. Krivelevich, S. Litsyn, and D. Ron. Testing Reed-Muller Codes. *IEEE Transactions on Information Theory*, 51(11):4032–4039, 2005.

3 Roksana Baleshzar, Meiram Murzabulatov, Ramesh Krishnan S. Pallavoor, and Sofya Raskhodnikova. Testing unateness of real-valued functions. *CoRR*, abs/1608.07652, 2016.

4 A. Belovs and E. Blais. A polynomial lower bound for testing monotonicity. In *Proceedings of the 48th ACM Symposium on Theory of Computing*, pages 1021–1032, 2016.

5 Arthur J. Bernstein. Maximally connected arrays on the n-cube. *SIAM J. Appl. Math.*, 15(6):1485–1489, 1967.

6 Arnab Bhattacharyya, Swastik Kopparty, Grant Schoenebeck, Madhu Sudan, and David Zuckerman. Optimal testing of reed-muller codes. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010*, pages 488–497, 2010.

7 Eric Blais. Improved bounds for testing juntas. In *Proc. RANDOM*, pages 317–330, 2008.

8 Eric Blais. Testing juntas nearly optimally. In *Proc. 41st Annual ACM Symposium on Theory of Computing (STOC)*, pages 151–158, 2009. doi:10.1145/1536414.1536437.

9 Eric Blais, Joshua Brody, and Kevin Matulef. Property testing lower bounds via communication complexity. In *CCC*, pages 210–220, 2011.

10 Eric Blais and Daniel M. Kane. Tight bounds for testing k-linearity. In *RANDOM*, pages 435–446, 2012.

11 M. Blum, M. Luby, and R. Rubinfeld. Self-testing/correcting with applications to numerical problems. *Journal of Computer and System Sciences*, 47:549–595, 1993. Earlier version in STOC'90.

12 Harry Buhrman, David García-Soriano, Arie Matsliah, and Ronald de Wolf. The non-adaptive query complexity of testing k-parities. *Chicago Journal of Theoretical Computer Science*, 2013, 2013.

13 Deeparnab Chakrabarty and C. Seshadhri. A $o(n)$ monotonicity tester for boolean functions over the hypercube. In *Proceedings of the 45th ACM Symposium on Theory of Computing*, pages 411–418, 2013.

14 Deeparnab Chakrabarty and C. Seshadhri. A $\widetilde{O}(n)$ non-adaptive tester for unateness. *CoRR*, abs/1608.06980, 2016.

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 47th ACM Symposium on Theory of Computing*, pages 519–528, 2015.

16 Xi Chen, Rocco A. Servedio, and Li-Yang Tan. New algorithms and lower bounds for testing monotonicity. In *Proceedings of the 55th IEEE Symposium on Foundations of Computer Science*, pages 286–295, 2014.

17 H. Chockler and D. Gutfreund. A lower bound for testing juntas. *Information Processing Letters*, 90(6):301–305, 2004.

18 I. Diakonikolas, H. Lee, K. Matulef, K. Onak, R. Rubinfeld, R. Servedio, and A. Wan. Testing for concise representations. In *Proc. 48th Ann. Symposium on Computer Science (FOCS)*, pages 549–558, 2007.

19 E. Fischer, G. Kindler, D. Ron, S. Safra, and A. Samorodnitsky. Testing juntas. *J. Computer & System Sciences*, 68(4):753–787, 2004.

20 E. Fischer, E. Lehman, I. Newman, S. Raskhodnikova, R. Rubinfeld, and A. Samorodnitsky. Monotonicity testing over general poset domains. In *Proc. 34th Annual ACM Symposium on the Theory of Computing*, pages 474–483, 2002.

21 Peter Frankl. On the trace of finite sets. *J. Comb. Theory, Ser. A*, 34(1):41–45, 1983.

22 O. Goldreich, editor. *Property Testing: Current Research and Surveys*. Springer, 2010. LNCS 6390.

23 O. Goldreich, S. Goldwasser, E. Lehman, D. Ron, and A. Samordinsky. Testing monotonicity. *Combinatorica*, 20(3):301–337, 2000.

24 P. Gopalan, R. O'Donnell, R. Servedio, A. Shpilka, and K. Wimmer. Testing Fourier dimensionality and sparsity. *SIAM J. on Computing*, 40(4):1075–1100, 2011.

25 Larry H. Harper. Optimal assignments of numbers to vertices. *SIAM J. Appl. Math.*, 12(1):131–135, 1964.

26 Sergiu Hart. A note on the edges of the n-cube. *Disc. Math.*, 14:157–163, 1976.

27 Subhash Khot, Dor Minzer, and Muli Safra. On monotonicity testing and boolean isoperimetric type theorems. In *Proceedings of the 56th Annual Symposium on Foundations of Computer Science*, pages 52–58, 2015.

28 Subhash Khot and Igor Shinkar. An o(n) queries adaptive tester for unateness. In *Approximation, Randomization, and Combinatorial Optimization Algorithms and Techniques*, 2016.

29 J. H. Lindsey. Assignment of numbers to vertices. *Amer. Math. Monthly*, 71:508–516, 1964.

30 K. Matulef, R. O'Donnell, R. Rubinfeld, and R. Servedio. Testing halfspaces. *SIAM J. on Comput.*, 39(5):2004–2047, 2010.

31 Kevin Matulef, Ryan O'Donnell, Ronitt Rubinfeld, and Rocco A. Servedio. Testing ± 1-weight halfspace. In *APPROX-RANDOM*, pages 646–657, 2009. doi:10.1007/978-3-642-03685-9_48.

32 M. Parnas, D. Ron, and A. Samorodnitsky. Testing Basic Boolean Formulae. *SIAM J. Disc. Math.*, 16:20–46, 2002.

33 D. Ron. Property Testing: A Learning Theory Perspective. *Foundations and Trends in Machine Learning*, 1(3):307–402, 2008.

34 D. Ron. Algorithmic and analysis techniques in property testing. *Foundations and Trends in Theoretical Computer Science*, 5:73–205, 2010.

35 D. Ron and R. Servedio. Exponentially improved algorithms and lower bounds for testing signed majorities. In *SODA*, pages 1319–1336, 2013.

36 Dana Ron and Gilad Tsur. Testing computability by width-two obdds. Technical Report 11(041), ECCC, 2011. available at http://eccc.hpi-web.de/report/2011/041/.

37 B. Roos. Binomial approximation to the Poisson binomial distribution: The Krawtchouk expansion. *Theory Probab. Appl.*, 45:328–344, 2000.

38 Rocco Servedio, Li-Yang Tan, and John Wright. Adaptivity helps for testing juntas. In *Proceedings of the 30th IEEE Conference on Computational Complexity*, pages 264–279, 2015. volume 33 of LIPIcs.

A Proof of Theorem 1 assuming Theorem 2

We prove the following claim in Appendix A.1.

▶ **Claim 17.** *Let $\epsilon(n)$ be a function that satisfies $2^{-n} \leq \epsilon(n) \leq 1/5$ for sufficiently large n. Then any non-adaptive algorithm that accepts the all-0 function with probability at least $5/6$ and rejects every function that is ϵ-far from $(n-1)$-juntas with probability at least $5/6$ must make $\Omega(1/\epsilon)$ queries.*

Next let $k(n)$ and $\epsilon(n)$ be the pair of functions from the statement of Theorem 1. We consider a sufficiently large n (letting $k = k(n)$ and $\epsilon = \epsilon(n)$ below) and separate the proof into two cases:

$$2^{-(2\alpha-1)k/(2\alpha)} \leq \epsilon \leq 1/6 \quad \text{and} \quad 2^{-n} \leq \epsilon < 2^{-(2\alpha-1)k/(2\alpha)}.$$

For the first case, if $k = O(1)$ then the bound we aim for is simply $\widetilde{\Omega}(1/\epsilon)$, which follows trivially from Claim 17 (since $k \leq \alpha n < n-1$ and the all-0 function is a k-junta). Otherwise we combine the following reduction with Theorem 2: any ϵ-tester for k-juntas over n-variable functions can be used to obtain an ϵ-tester for k-juntas over (k/α)-variable functions. This can be done by adding $n - k/\alpha$ dummy variables to any (k/α)-variable function to make the number of variables n (as $k \leq \alpha n$). The lower bound then follows from Theorem 2 since α is a constant. For the second case, the lower bound claimed in Theorem 1 is $\widetilde{\Omega}(1/\epsilon)$, which follows again from Claim 17. This concludes the proof of Theorem 1 given Theorem 2 and Claim 17. ◀

A.1 Proof of Claim 17

Let C be a sufficiently large constant. We prove Claim 17 by considering two cases:

$$\epsilon \geq \frac{C \log n}{2^n} \quad \text{and} \quad \epsilon < \frac{C \log n}{2^n}.$$

For the first case of $2^n \epsilon \geq C \log n$, we use $\mathcal{D}_1$ to denote the following distribution over n-variable Boolean functions: to draw $\boldsymbol{g} \sim \mathcal{D}_1$, independently for each $x \in \{0,1\}^n$ the value of $\boldsymbol{g}(x)$ is set to 0 with probability $1 - 3\epsilon$ (recall that $\epsilon \leq 1/5$) and 1 with probability 3ϵ.

We prove the following lemma for the distribution $\mathcal{D}_1$:

▶ **Lemma 18.** *With probability at least $1 - o(1)$, $\boldsymbol{g} \sim \mathcal{D}_1$ is ϵ-far from every $(n-1)$-junta.*

Proof. Note that every $(n-1)$-junta is such that for some $i \in [n]$, the function does not depend on the i-th variable; we refer to such a function as a type-i junta. An easy lower bound for the distance from a function g to all type-i juntas is the number of g-bichromatic edges $(x, x^{(i)})$ divided by 2^n. When $\boldsymbol{g} \sim \mathcal{D}_1$ each edge $(x, x^{(i)})$ is independently $\boldsymbol{g}$-bichromatic with probability $6\epsilon(1-3\epsilon) \geq 12\epsilon/5$ (as $\epsilon \leq 1/5$). Thus when $2^n\epsilon \geq C \log n$, the expected number of such edges is at least

$$2^{n-1} \cdot (12\epsilon/5) \geq (6/5) \cdot 2^n\epsilon \geq (6/5) \cdot C \log n.$$

Using a Chernoff bound, the probability of having fewer than $2^n\epsilon$ bichromatic edges along direction i is at most $1/n^2$ when C is sufficiently large. The lemma follows from a union bound over i. ◀

As a result, when $2^n\epsilon \geq C \log n$, if $\mathcal{A}$ is a non-adaptive algorithm with the property described in Claim 17, then $\mathcal{A}$ must satisfy

$$\mathbf{Pr}\left[\mathcal{A} \text{ accepts the all-0 function}\right] - \mathop{\mathbf{Pr}}_{\boldsymbol{g}\sim\mathcal{D}_1}\left[\mathcal{A} \text{ accepts } \boldsymbol{g}\right] \geq 2/3 - o(1).$$

But any such non-adaptive algorithm must make $\Omega(1/\epsilon)$ queries as otherwise with high probability all of its queries to $\boldsymbol{g} \sim \mathcal{D}_1$ would be answered 0, and hence its behavior would be the same as if it were running on the all-0 function.

Finally we work on the case when $1 \leq 2^n\epsilon = O(\log n)$. The proof is the same except that we let $\boldsymbol{g}$ be drawn from $\mathcal{D}_2$, which we define to be the distribution where all entries of $\boldsymbol{g} \sim \mathcal{D}_2$ are 0 except for exactly $2^n\epsilon$ of them picked uniformly at random. The claim follows from the following lemma:

▶ **Lemma 19.** *With probability at least $1 - o(1)$, $\boldsymbol{g} \sim \mathcal{D}_2$ is ϵ-far from every $(n-1)$-junta.*

Proof. This follows from the observation that, with probability $1-o(1)$, no two points picked form an edge. When this happens, we have $2^n\epsilon$ bichromatic edges along the ith direction for all i. ◀

An Adaptivity Hierarchy Theorem for Property Testing

Clément L. Canonne[1] and Tom Gur[2]

1 **Columbia University, New York, NY, USA**
ccanonne@cs.columbia.edu
2 **UC Berkeley, Berkeley, CA, USA**
tom.gur@berkeley.edu

Abstract

Adaptivity is known to play a crucial role in property testing. In particular, there exist properties for which there is an exponential gap between the power of *adaptive* testing algorithms, wherein each query may be determined by the answers received to prior queries, and their *non-adaptive* counterparts, in which all queries are independent of answers obtained from previous queries.

In this work, we investigate the role of adaptivity in property testing at a finer level. We first quantify the degree of adaptivity of a testing algorithm by considering the number of "rounds of adaptivity" it uses. More accurately, we say that a tester is k-(round) adaptive if it makes queries in $k+1$ rounds, where the queries in the i'th round may depend on the answers obtained in the previous $i-1$ rounds. Then, we ask the following question:

> *Does the power of testing algorithms smoothly grow with the number of rounds of adaptivity?*

We provide a positive answer to the foregoing question by proving an adaptivity hierarchy theorem for property testing. Specifically, our main result shows that for every $n \in \mathbb{N}$ and $0 \leq k \leq n^{0.99}$ there exists a property $\mathcal{P}_{n,k}$ of functions for which (1) there exists a k-adaptive tester for $\mathcal{P}_{n,k}$ with query complexity $\tilde{O}(k)$, yet (2) any $(k-1)$-adaptive tester for $\mathcal{P}_{n,k}$ must make $\Omega(n)$ queries. In addition, we show that such a qualitative adaptivity hierarchy can be witnessed for testing natural properties of graphs.

1998 ACM Subject Classification F.1.3 [Computation by Abstract Devices] Complexity Measures and Classes

Keywords and phrases Property Testing, Coding Theory, Hierarchy Theorems

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.27

1 Introduction

The study of property testing, initiated by Rubinfeld and Sudan [36] and Goldreich, Goldwasser and Ron [21], has attracted significant attention in the last two decades (see, e.g., recent books [18, 20, 5] and surveys [32, 33, 15]). Loosely speaking, property testers are highly efficient randomized algorithms (typically running in sublinear time) that solve approximate decision problems, while only inspecting a tiny fraction of their inputs. More accurately, an ε-tester $\mathcal{T}$ for property $\mathcal{P}$ is a randomized algorithm that, given query access to an input x, decides whether $x \in \mathcal{P}$ or x is ε-far (say, in Hamming distance) from $\mathcal{P}$. The query complexity of $\mathcal{T}$ is then the number of queries it makes to x.

In general, a testing algorithm may select its queries adaptively such that the i'th query is determined by the answers to the previous $i-1$ queries, in which case it is said to be an

32nd Computational Complexity Conference (CCC 2017).
Editor: Ryan O'Donnell; Article No. 27; pp. 27:1–27:25

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

adaptive tester. However, in many natural cases, testers may actually determine their queries solely based on their randomness (and input length), without any dependency on answers to previous queries; a tester that satisfies this condition is called a *non-adaptive tester*. A natural question, which commonly arises in query-based models, is whether the ability to make adaptive queries can significantly affect the query complexity.

Adaptive queries can be easily emulated at the cost of a large blowup in query complexity (exponential in the number of queries). More accurately, any q-query adaptive tester for a property of objects represented by functions $f\colon D \to R$ can be emulated by an $|R|^q$-query non-adaptive tester (see e.g., [20, Section 1.5]). While for certain types of properties and models – e.g., linear properties [4] and properties in the dense graph model [25] – one has better emulations which come with little or no overhead, such efficient emulations cannot exist for all properties. As was shown by Raskhodnikova and Smith [31], in the bounded-degree graph model [23] there is a large chasm between the adaptive and non-adaptive query complexities of testing many natural graph properties. In particular, any property over bounded-degree graphs with n vertices, which is not determined by the *vertex degree distribution*,[1] requires $\Omega(\sqrt{n})$ queries to test non-adaptively, whereas many such properties (e.g., triangle-freeness and connectivity) have ε-testers with query complexity $\mathrm{poly}(1/\varepsilon)$.

In this work, we investigate the role of adaptivity in property testing at a finer level. Rather than considering the extreme cases of fully adaptive testers versus completely non-adaptive testers, we consider testers with various levels of restricted adaptivity and ask the following question:

> *Can the power of testers gradually grow with the "amount" of adaptivity they are allowed to use?*

Besides the sheer theoretical interest of understanding the role of adaptivity in property testing, a motivation for this question comes from the *constraints* that come with adaptive algorithms, which may counterbalance the apparent gain in efficiency. Indeed, non-adaptive algorithms (or at least those which only use a small number of adaptive "stages") may be preferred in practice to their adaptive counterparts, in spite of the larger number of queries they make. The reason for this preference is the significant gains obtained by being able to make many queries *in parallel*: when each query is an experiment which, while relatively cheap by itself, may take several hours, assessing the trade-off between rounds of adaptivity and total number of queries becomes crucial. An archetypal example where such considerations prevail is the (different) setting of group testing (see e.g. [17, Section 1.2]).

To answer the foregoing question, we shall first need to give a precise definition for the "amount" of adaptivity that a tester uses. To this end, it is natural to consider the number of "rounds of adaptivity" used by a tester.[2] More precisely, we say that a tester is *k-round-adaptive* if it generates and makes queries in $k+1$ rounds, where in the i'th round the tester queries a set of locations Q_i that may depend on the answers to queries in $Q_0, \ldots, Q_{i-1}$, obtained in previous rounds. We will quantify the "amount" of adaptivity that a tester uses by the number of rounds of adaptivity that it uses. Equipped with the notion of round adaptivity, we can proceed to present our results.

---

[1] Loosely speaking, a property $\mathcal{P}$ of bounded-degree graphs is not determined by the vertex degree distribution if there exist two graphs, $G_1 \in \mathcal{P}$ and G_2 that is "far" from $\mathcal{P}$, such that the vertices of G_1 and G_2 have the same degrees.

[2] We also consider an alternative notion of *tail adaptivity*, which roughly speaking refers to testers that first make a large number of non-adaptive queries and subsequently make a bounded number of adaptive queries. See Section 3 for details regarding how these two notions relate.

1.1 Our Results

Our main result provides a positive answer to the foregoing question by showing an adaptivity hierarchy theorem for property testing; that is, we show a family of properties $\{\mathcal{P}_k\}_k$ such that for every k, the property $\mathcal{P}_k$ is "easy" for k-adaptive testers and "hard" for $(k-1)$-adaptive testers.

▶ **Theorem 1** (Informally stated (see Theorem 5)). *For every $n \in \mathbb{N}$ and $0 \leq k \leq n^{0.33}$ there is a property $\mathcal{P}_{n,k}$ of strings over $\mathbb{F}_n$ (of length that is nearly linear in n) such that:*
1. *there exists a k-round-adaptive tester for $\mathcal{P}_{n,k}$ with query complexity $\tilde{O}(k)$, yet*
2. *any $(k-1)$-round-adaptive tester for $\mathcal{P}_{n,k}$ must make $\tilde{\Omega}(n/k^2)$ queries.*

The above theorem relies on an arguably contrived family of properties, which was specifically tailored towards maximizing the separations; hence, one may wonder whether such strong separations also hold for more natural properties. As we show below, this is indeed the case: namely, we establish another adaptivity hierarchy theorem that, albeit weaker than Theorem 1, applies to the well-studied natural problem of testing k-cycle freeness in the bounded-degree graph model (see Section 5.1 for definitions).

▶ **Theorem 2.** *Let $k \in \mathbb{N}$ be a constant. Then,*
1. *there exists a k-round-adaptive tester with query complexity $O(1/\varepsilon)$ for $(2k+1)$-cycle freeness in the bounded-degree graph model; yet*
2. *any $(k-1)$-round-adaptive tester for $(2k+1)$-cycle freeness in the bounded-degree graph model must make $\Omega(\sqrt{n})$ queries, where n is the number of vertices in the graph.*

Lifting Query Complexity Bounds to Property Testing. Notably, the proof of Theorem 1 relies on a technique that allows us to "lift" *query complexity* bounds to property testing, *via* the use of error-correcting codes that admit a strong form of local testability as well as a relaxed form of local decodability. We believe that this framework, which we detail in Section 4.4, is of interest in its own right, and will find further applications in property testing.

We conclude this section by posing two open problems that naturally arise from our work.

▶ **Open Problem 1** (One property to rule them all). *Does there exist an adaptivity hierarchy with respect to a* single *property? That is, for any m and all sufficiently large n, is there a property $\mathcal{P}$ of elements of size n, and $q_1 > \ldots > q_m$ (m "levels" of hierarchy) such that for every $k \in [m]$ there exists a k-adaptive tester for $\mathcal{P}$ with query complexity q_k, yet every $(k-1)$-adaptive tester must make $\omega(q_k)$ queries to test $\mathcal{P}$?*

▶ **Open Problem 2** (Au naturel is just as good). *Does there exist a family of* natural *properties which exhibits an adaptivity hierarchy with separations as strong as in Theorem 1?*

1.2 Previous Work

As previously mentioned, the role of adaptivity in property testing has been the focus of several works before. It is well known that for any property of Boolean functions, there exists at most an exponential gap between adaptive and non-adaptive testers: any (adaptive) q-query testing algorithm for a property $\mathcal{P}$ of n-variate Boolean functions can be simulated by a non-adaptive tester with query complexity $2^q - 1$. Further, such gaps are known to exist for some natural properties, such as read-once width-2 OBDDs [35, 12] and signed majorities [28, 34] (importantly, there also exist cases where adaptivity is known *not* to

help [11, 4]). Another prominent example of a class of Boolean functions where adaptivity is known to help is that of k-juntas [8, 7, 38, 16], which can be tested adaptively with $\tilde{O}(k)$ queries, yet for which the non-adaptive query complexity is $\tilde{\Theta}\left(k^{3/2}\right)$.

Of course, the Boolean function setting is not the only one: in the dense graph model, it is known that while adaptivity can help [24], it will be at most by a quadratic factor [2, 25]: that is, every graph property testable (adaptively) with q queries has an $O\left(q^2\right)$-query non-adaptive tester. This is no longer the case in the bounded-degree model, however; where Raskhodnikova and Smith showed that there exist many properties which can be tested adaptively with a constant number of queries, but for which any non-adaptive tester must have query complexity $\Omega(\sqrt{n})$ [31].[3]

However, all these results, even when they establish cases where adaptivity does help, leave open the question of *how much* adaptivity is needed for this to happen. In particular, for the case of properties of Boolean functions, many known adaptive testers which outperforms their non-adaptive counterpart do so, at some level, by conducting a binary search of some sort (see, e.g., [8, 35, 34]) and thus comes inherently with a logarithmic numbers of "adaptive rounds."

Our proof of Theorem 1 relies on a connection between the property testing and linear decision tree models. Although many of the ingredients we use are new, the connection itself is not and was first observed in [39] (see also [6] for a slightly different connection between property testing and parity decision trees).

Adaptivity in other settings. We remark that the notion of round complexity in communication complexity and interactive proof systems is somewhat analogous to that of round adaptivity, since in those models each round of communication or interaction allows the parties to adapt their strategies. Moreover, a round complexity hierarchy is known for communication complexity [29] and interactive proofs of proximity [26]. Finally, we also mention that the role of the number of adaptive measurements used by sparse recovery algorithms was shown to be very significant [27].

Organization

In Section 2 we provide the preliminaries required for the technical sections. In Section 3 we provide a precise definition for testers with bounded adaptivity. In Section 4 we prove our main result, which is a strong adaptivity hierarchy theorem for a property of functions. In Section 5 we prove an adaptivity hierarchy theorem with respect to a natural property of graphs. Finally, in Section 6 we discuss adaptivity round reductions, as well as a connection to communication complexity, and the relation between round and tail adaptivity.

2 Preliminaries

We begin with standard notations:

- We denote the *relative Hamming distance*, over alphabet Σ, between two vectors $x \in \Sigma^n$ and $y \in \Sigma^n$ by $\operatorname{dist}(x,y) \stackrel{\text{def}}{=} |\{x_i \neq y_i : i \in [n]\}|/n$. If $\operatorname{dist}(x,y) \leq \varepsilon$, we say that x is ε-close to y, and otherwise we say that x is ε-far from y. Similarly, we denote the

[3] We remark that in the bounded-degree model, algorithms typically rely on random walks or breadth-first searches. In this case the depth of the walks or searches, which may be smaller than the query complexity, tends to determine the adaptivity.

relative distance of x from a non-empty set $S \subseteq \Sigma^n$ by $\mathrm{dist}(x,S) \stackrel{\text{def}}{=} \min_{y \in S} \mathrm{dist}(x,y)$. If $\mathrm{dist}(x,S) \leq \varepsilon$, we say that x is ε-close to S, and otherwise we say that x is ε-far from S.

- We denote by $A^x(y)$ the output of algorithm A given direct access to input y and oracle access to string x. Given two interactive machines A and B, we denote by $(A^x, B(y))(z)$ the output of A when interacting with B, where A (respectively, B) is given oracle access to x (respectively, direct access to y) and both parties have direct access to z. Throughout this work, probabilistic expressions that involve a randomized algorithm A are taken over the inner randomness of A (e.g., when we write $\Pr[A^x(y) = z]$, the probability is taken over the coin tosses of A).
- We use the notations $\tilde{O}(f), \tilde{\Omega}(f)$ to hide polylogarithmic dependencies on the argument, i.e. for expressions of the form $O(f \log^c f)$ and $\Omega(f \log^c f)$ (for some absolute constant c). Finally, all our logarithms are in base 2.

Integrality. For simplicity of notation, we hereafter use the convention that all (relevant) integer parameters that are stated as real numbers are implicitly rounded to the closest integer.

Uniformity. To facilitate notation, throughout this work we define all algorithms *non-uniformly*; that is, we fix an integer $n \in \mathbb{N}$ and restrict the algorithms to inputs of length n. Despite fixing n, we view it as a generic parameter and allow ourselves to write asymptotic expressions such as $O(n)$. We remark that while our results are proved in terms of non-uniform algorithms, they can be extended to the uniform setting in a straightforward manner.

3 The Definition of Testers with Bounded Adaptivity

In this section, we provide a formal abstraction that captures the notion of *bounded adaptivity* within the framework of property testing. We define two notions of bounded adaptivity: (1) *round-adaptivity*, which refers to algorithms that are allowed to make a bounded number of "batches" of queries, where the queries in each batch may depend on the answers to previous batches; (2) *tail-adaptivity*, which refers to algorithms that first make a large number of non-adaptive queries and subsequently make a bounded number of adaptive queries.

We remark that while tail-adaptivity can be easily emulated via round-adaptivity, the converse does *not* hold. Indeed, in Section 6.3 we show that round-adaptive testers can be much more powerful than tail-adaptive testers. Nonetheless, our lower bounds hold for the stronger round-adaptivity notion, whereas out upper bounds hold for the more restrictive tail-adaptivity.

▶ **Definition 3** (Round-Adaptive Testing Algorithms)**.** Let $[n]$ be a domain of cardinality n, and let $k, q \leq n$. A randomized algorithm is said to be a (k,q)-*round-adaptive* tester for a property $\mathcal{P} \subseteq 2^{[n]}$, if, on proximity parameter $\varepsilon \in (0,1]$ and granted query access to a function $f\colon [n] \to \{0,1\}$, the following holds.

1. Query Generation: The algorithm proceeds in $k+1$ rounds, such that at round $\ell \geq 0$, it produces a set of queries $Q_\ell \stackrel{\text{def}}{=} \{x^{(\ell),1}, \ldots, x^{(\ell),|Q_\ell|}\} \subseteq [n]$ (possibly empty), based on its own internal randomness and the answers to the previous sets of queries $Q_0, \ldots, Q_{\ell-1}$, and receives $f(Q_\ell) = \{f(x^{(\ell),1}), \ldots, f(x^{(\ell),|Q_\ell|})\}$;
2. Completeness: If $f \in \mathcal{P}$, then the algorithm outputs accept with probability at least $2/3$;
3. Soundness: If $\mathrm{dist}(f, \mathcal{P}) > \varepsilon$, then the algorithm outputs reject with probability at least $2/3$.

The *query complexity* q of the tester is the total number of queries made to f, i.e., $q = \sum_{\ell=0}^{k} |Q_\ell|$. If the algorithm returns accept with probability one whenever $f \in \mathcal{P}$, it is said to have *one-sided* error (otherwise, it has *two-sided* error). We will sometimes refer to a tester with respect to proximity parameter ε as an ε-tester.

▶ Remark (On amplification). We note that, as usual in property testing, the probability of success can be amplified by repetition to any $1-\delta$, at the price of an $O(\log(1/\delta))$ factor in the query complexity. Crucially, this can be done with no increase in the number of adaptive rounds: while repetition would naïvely multiply both q *and* k by this factor, one can avoid the latter by running the $O(\log(1/\delta))$ independent copies of the algorithm in parallel, instead of sequentially.

▶ **Definition 4** (Tail-Adaptive Testing Algorithms)**.** Let $[n]$ be a domain of cardinality n, and let $k, q \leq n$. A randomized algorithm is said to be a (k,q)-*tail-adaptive* tester for a property $\mathcal{P} \subseteq 2^{[n]}$, if, on proximity parameter $\varepsilon \in (0,1]$, error parameter $\delta \in (0,1]$, and granted query access to a function $f\colon [n] \to \{0,1\}$, the following holds.

1. Query Generation: The algorithm proceeds in $k+1$ rounds, such that in the first round, it produces a set of queries $Q \stackrel{\text{def}}{=} \{x^{(0),1}, \ldots, x^{(0),|Q|}\} \subseteq [n]$ (possibly empty), based on its own internal randomness; and receives $f(Q) = \{f(x^{(0),1}), \ldots, f(x^{(0),|Q|})\}$; then it makes, over the next k rounds, k adaptive queries to f, denoted $x^{(1)}, \ldots, x^{(k)}$;
2. Completeness: If $f \in \mathcal{P}$, then the algorithm outputs accept with probability at least $1-\delta$;
3. Soundness: If $\mathrm{dist}(f, \mathcal{P}) > \varepsilon$, then the algorithm outputs reject with probability at least $1-\delta$.

The *query complexity* q of the tester is the total number of queries made to f, i.e., $q = |Q| + k$. If the algorithm returns accept with probability one whenever $f \in \mathcal{P}$, it is said to be *one-sided* (otherwise, it is *two-sided*).

▶ Remark (On (lack of) amplification). Unlike the round-adaptive algorithms, tail-adaptive testing algorithms do not enjoy a simple success amplification procedure which would leave unchanged the adaptivity parameter, only affecting the query complexity. This is the reason why the success probability δ is explicitly mentioned in Definition 4.

4 A Strong Adaptivity Hierarchy

In this section we prove the adaptivity hierarchy theorem, which shows that, loosely speaking, up to a nearly linear threshold, each additional round of adaptivity can significantly augment the power of testing algorithms.

▶ **Theorem 5** (Adaptivity Hierarchy Theorem)**.** *Fix any $\alpha \in (0,1)$. There exists a constant $\beta \in (0,1)$ such that, for every $n \in \mathbb{N}$, the following holds. For every integer $0 \leq k \leq n^\beta$, there exists a property $\mathcal{P}_k \subseteq \mathbb{F}_n^{n^{1+\alpha}}$ such that, for any constant $\varepsilon \in (0,1]$,*

1. *there exists a $(k, \tilde{O}(k))$-round-adaptive (one-sided) tester for $\mathcal{P}_k$; yet*
2. *any $(k-1,q)$-round-adaptive (two-sided) tester for $\mathcal{P}_k$ must satisfy $q = \tilde{\Omega}\big(n/k^2\big)$.*

We remark that, in fact, the algorithm shown in the first item of Theorem 5 also gives an upper bound for the more restricted model of *tail adaptivity*. Specifically, for every k there also exists an $(O(k), \tilde{O}(k))$-tail-adaptive (one-sided) tester for $\mathcal{P}_k$. Since a $(k-1,q)$-round-adaptive lower bound implies a $(k-1,q)$-tail-adaptive lower bound (see discussion in Section 3), this implies an adaptivity hierarchy (albeit slightly weaker than in Theorem 5) with respect to tail-adaptive testers.

To prove Theorem 5, we use a technique that allows us to "lift" bounds on decision tree complexity to the setting of property testing, relying on error-correcting codes with local properties. In more detail, we shall begin by proving an analogous version of Theorem 5 for the (linear) decision tree model, which is an elementary computation model that is significantly easier to analyze than property testing (in particular, it deals with *exact* decision problems, rather than *approximate* decision problems). Then, we shall consider an encoded version of the decision tree problem, where the encoding is via a code that admits strong local testability and a relaxed form of local decodability. Capitalizing on the foregoing properties of the code, we show transference lemmas that allow us to "lift" our bounds on the decision tree problem to bounds on the complexity of the encoded problem in the property testing model. (We believe that this "lifting" technique may find further uses in property testing, by allowing one to show results in the simpler decision tree problem before carrying them over to property testing.)

We begin by describing the decision tree problem with respect to which we prove the hierarchy theorem. Hereafter we assume, without loss of generality,[4] that n is a prime number, and consider $\mathbb{F}_n$, the field of order n. We will consider the following sequence of "k-iterated address" functions $(f_k)_{k\geq 0}$ from $\mathbb{F}_n^n$ to $\{0,1\}$, which will in turn lead to the definition of the properties $(\mathcal{P}_k)_{k\geq 0}$ that we use to show the hierarchy theorem. Loosely speaking, f_k receives a vector x of n pointers (indices in $[n]$) and indicates whether when jumping from pointer to pointer k times, starting from an arbitrarily predetermined pointer, we reach a location in which x takes an even value.

To formally define the foregoing functions, first consider $g\colon \mathbb{F}_n^n \times \mathbb{F}_n \to \mathbb{F}_n$ given by $g(x,a) = x_{a+1}$; that is, g returns the coordinate of $x \in \mathbb{F}_n^n$ "pointed to" by $a \in \{0,\dots,n-1\}$. Based on this, we define the iterated versions of g, $g_0,\dots,g_n,\dots : \mathbb{F}_n^n \to \mathbb{F}_n$, as

$$\begin{aligned} g_0(x) &= g(x,0)\,, \\ g_k(x) &= g(x, g_{k-1}(x))\,. \end{aligned} \qquad (k \geq 1)$$

Finally, we define the *k-iterated address* function $f_k\colon \mathbb{F}_n^n \to \{0,1\}$ by

$$f_k(x) = \mathbb{1}_{\{g_k(x) \text{ even}\}} = \begin{cases} 1 & \text{if } g_k(x) \text{ is even,} \\ 0 & \text{otherwise.} \end{cases}$$

(For instance, $f_0(x) = 1$ if and only if x_1 is even; and $f_1(x) = 1$ if and only if the coordinate of x pointed to by x_1, that is x_{x_1+1}, is even.) We proceed to describe the outline of the proof of Theorem 5.

4.1 High-Level Overview

Broadly speaking, our roadmap for proving Theorem 5 consists of two main steps:

1. We first consider the adaptivity hierarchy question in the setting of randomized *decision tree* (DT) complexity (see Section 4.2). We can view a randomized DT for computing a function f as a probabilistic algorithm that is given query access to an input x and is required to output $f(x)$ with high probability. Adapting the definition of round adaptivity (Definition 3) in the natural way to decision trees, we will prove the randomized DT analogue of our adaptivity hierarchy theorem, using the foregoing family of address

[4] If n is not prime, we choose a prime p such that $n \leq p \leq 2p$, and use standard padding techniques.

functions $(f_k)_{k\geq 0}$. Namely, we prove that for any $k \geq 0$ with $k = o(n)$, it holds that (i) f_k can be computed by an algorithm making $k+1$ queries, in k adaptive rounds; but (ii) any algorithm using only $k-1$ rounds of adaptivity must make $\tilde{\Omega}\big(n/k^2\big)$ queries.

2. We then show a bidirectional connection between *adaptivity-bounded* randomized DT and property testers, which extends the connection observed by Tell [39]. This allows us to "lift" the DT adaptivity hierarchy theorem to property testing. Specifically, we provide two blackbox reductions between the DT problem of computing function f and property testing for a related property $\mathcal{P}_f$, which preserve both the number of adaptive rounds and (roughly) the number of queries. We remark these reductions strongly rely on high-rate codes that exhibit both strong local testability and relaxed local decodability.

The caveat with the above is that to "lift" DT lower bounds to testing algorithms via our methodology, we actually need to show lower bounds on a stronger model of DT (this stems from the reductions of the second item, in which we will encode the input via linear codes, requiring the DT algorithm to compute coordinates of this encoding).

Hence, we will actually work in the *linear decision tree* (LDT) model, wherein the algorithm is allowed to query any linear combination (over $\mathbb{F}_n$) of the coordinates, instead of only querying individual coordinates. (We note that in the case of $\mathbb{F}_2$, this corresponds to the *parity decision tree* model.) That is, we will proceed as follows:

1. (L)DT hierarchy: show that for any $k \geq 0$, the function f_k (i) can be computed by an efficient $(k, O(k))$-round-adaptive (deterministic) DT algorithm, but (ii) does not admit any $(k-1, o(n))$-round-adaptive (randomized, two-sided) LDT algorithm;
2. Transference lemmas: Show that for any function $f\colon \mathbb{F}_n^n \to \mathbb{F}_n$, there exists a property $\mathcal{C}_f \subseteq \mathbb{F}_n^{m(n)}$ such that, for any $k \geq 0$,
   a. a (k,q)-round-adaptive testing algorithm for $\mathcal{C}_f$ implies a (k,q)-round-adaptive LDT algorithm for f (Lemma 14).
   b. a (k,q)-round-adaptive DT algorithm for f implies a $(k, \tilde{O}(q))$-round-adaptive testing algorithm for $\mathcal{C}_f$ (Lemma 15).

Combining the items above will directly imply our hierarchy theorem for property testing (Theorem 5):

Proof. Proof of Theorem 5 The upper bound 1 follows immediately from Claim 7 and Lemma 15, while combining Lemma 8 and Lemma 14 establishes the lower bound 2. ◀

Organization for the rest of the section. In Section 4.2, we define the decision tree models and complexities that we shall need. Then, in Section 4.3, we prove the adaptivity hierarchy theorem for randomized (linear) decision trees. Finally, in Section 4.4 we prove the transference lemmas that allow us to lift the foregoing hierarchy theorem to the property testing framework.

4.2 Decision Tree Zoo

We shall need to extend the definitions of several different types of decision tree algorithms (see [13] for an extensive survey of decision tree complexity) to the setting of bounded adaptivity.

Recall that a deterministic decision tree is a model of computation for computing a function $f\colon [n]^n \to [n]$. The decision tree is a rooted ordered $|[n]|$-ary tree. Each internal vertex of the tree is labeled with a value $i \in \{1, \ldots, n\}$ and the leaves of the tree are labeled with the elements in $[n]$. Given an input $x \in [n]^n$, the decision tree is recursively evaluated by

choosing to recurse on the i'th subtree in the j'th level if and only if $x_j = i$. Once a leaf is reached, we output the label of that leaf and halt.

Equivalently, we can view deterministic decision trees as algorithms that get oracle access to an input $x \in [n]^n$, then adaptively make queries to x, to the end of computing $f(x)$. (Note that the j'th query corresponds to the j'th layer of the corresponding decision tree, and that the different vertices in the j'th layer represent the choices of the next queries, with respect to the answers obtained for previous queries). We define the *deterministic decision tree complexity* of a function f to be the minimal number of queries a deterministic decision tree algorithm needs to make to compute f in the worst case.[5]

Taking the algorithmic perspective, we define k-round-adaptive deterministic decision tree algorithms as algorithms that generate their queries in k rounds, where queries in each round may depend on queries from previous rounds. The extension of the foregoing definition to *randomized* decision tree algorithms is done in the natural way, by allowing the algorithm to toss random coins and succeed with high probability (say, 2/3) in computing $f(x)$. Finally, we shall also extend the definition to linear decision trees, which are decision trees algorithms wherein each query is a linear combination of the elements of the domain. We remark that linear decision trees can be thought of as generalizing both *parity decision trees* and *algebraic query complexity algorithms* [1].

More accurately, the aforementioned notions are defined below. We provide the definition of the most general model and derive the more restricted models as special cases.

▶ **Definition 6** (Round-Adaptive Decision Tree Algorithms)**.** Let $\mathbb{F}$ be a finite field of cardinality n, and let $k, q \leq n$. A (randomized) algorithm D is said to be a (k, q)-*round-adaptive* (linear) decision tree algorithm for computing a function $f\colon \mathbb{F}^n \to \mathbb{F}$ if, granted query access to a string $x \in \mathbb{F}^n$, the following holds.

1. Query Generation: The algorithm proceeds in $k+1$ rounds, such that at round $\ell \geq 0$, it produces a set of (linear) queries $Q_\ell \stackrel{\text{def}}{=} \{L_{\ell,1}, \ldots, L_{\ell,|Q_\ell|}\}$, where $L_{\ell,j} \in \mathbb{F}^n$ specifies a linear combination, based on its internal randomness and the answers to the previous sets of queries $Q_0, \ldots, Q_{\ell-1}$, and receives the answers $\langle L_{\ell,1}, x\rangle, \ldots, \langle L_{\ell,|Q_\ell|}, x\rangle$.

2. Computation: The algorithm computes $f(x)$ with high probability using the answers it received in all k rounds; that is, $\Pr[D^x = f(x)] \geq 2/3$.

The *query complexity* q of the tester is the total number of (linear) queries made to f, i.e., $q = \sum_{\ell=0}^{k} |Q_\ell|$. The *randomized* (k, q)*-round-adaptive linear decision tree complexity* of a function f, denoted $R_k^{\oplus}(f)$, is the minimal query complexity for a (k, q)-round-adaptive randomized linear decision tree algorithm that computes f.

If for all $\ell \in [k+1]$ and $j \in [|Q_\ell|]$ the linear combination $L_{\ell,j}$ only includes a single element (i.e., $L_{\ell,j}$ only has a single non-zero entry), we say that D is a *randomized* (k, q)*-round-adaptive decision tree algorithm complexity*, and denote its corresponding complexity by $R_k(f)$. If, in addition, the algorithm does not toss any random coins and succeeds with probability 1, we say that D is a *deterministic* (k, q)*-round-adaptive decision tree algorithm complexity*, and denote its corresponding complexity by $D_k(f)$.

[5] We remark that this definition corresponds to the depth the of decision tree, and not to the number of vertices or edges in the tree.

4.3 Decision Tree Hierarchy: Some Things Only Adaptivity Can Address

We first establish the upper bound part of our adaptivity hierarchy theorem for DT, which follows immediately from the construction.

▶ **Claim 7.** *For every $k \geq 0$, there exists a $(k, k+1)$-round-adaptive (deterministic) DT algorithm which computes f_k; that is, $D_k(f_k) \leq k+1$.*

Proof. The algorithm is straightforward: on input $x \in \mathbb{F}_n^n$, it sequentially queries $x_1 = g_0(x)$, $x_{g_0(x)+1} = g_1(x)$, …, $x_{g_{k-1}(x)+1} = g_k(x)$; and returns 1 if $g_k(x)$ is even, and 0 otherwise. By definition of f_k, this always correctly computes the function, is deterministic, and clearly satisfies the definition of a $(k, k+1)$-round-adaptive DT algorithm. ◀

We proceed to show the lower bound part of our adaptivity hierarchy theorem for DT, which is proven via a reduction from communication complexity.

▶ **Lemma 8.** *There exists an absolute constant $c > 0$ such that the following holds. For every $0 \leq k \leq c\left(\frac{n}{\log n}\right)^{1/3}$, there is no $(k, o(n/(k^2 \log n)))$-round-adaptive (randomized) LDT algorithm which computes f_{k+1}; that is, $R_k^{\oplus}(f_{k+1}) = \Omega\big(n/(k^2 \log n)\big)$.*

Proof. We will reduce to the computation of f_{k+1} (in k rounds of adaptivity) a related k-round two-party randomized communication complexity problem, the "pointer-following" problem introduced by Papadimitriou and Sipser [30], and conclude by invoking the lower bound of Nisan and Wigderson [29] on this problem.

This communication complexity problem between two computationally unbounded players, Alice and Bob, is defined as follows. Let V_A and V_B be two disjoint sets of cardinality $n/2$, and let $v_0 \in V_A$ be a fixed element known to both players. The input is a pair of functions (χ_A, χ_B), where $\chi_A \colon V_A \to V_B$ and $\chi_B \colon V_B \to V_A$. Alice and Bob are given χ_A and χ_B respectively, as well as a common random string, and their goal is to compute $\pi_k(\chi_A, \chi_B) \stackrel{\text{def}}{=} \chi^{(k)}(v_0)$ with high probability, where $\chi^{(\ell)}$ is the ℓ-iterate of the function χ:

$$\chi \colon V_A \cup V_B \to V_A \cup V_B$$
$$v \mapsto \begin{cases} \chi_A(v) & v \in V_A \\ \chi_B(v) & v \in V_B. \end{cases}$$

(In other terms, one can see the communication problem as Alice and Bob sharing the edges of a bipartite directed graph where each node has out-degree exactly one, and the goal is to find at which vertex the path of length k starting at a prespecified vertex v_0, on Alice's side, ends.)

We will rely on the following lower bound on the k-round, randomized (public-coin) version of this problem.

▶ **Theorem 9** ([29], rephrased)**.** *Any k-round randomized communication protocol for the "pointer-following" problem, in which Bob sends the first message, must have total communication complexity $\Omega\big(\frac{n}{k^2} - k \log n\big)$, even to only compute a single bit of $\pi_k(\chi_A, \chi_B)$ with probability at least $2/3$.*

Note that as long as $k \ll \left(\frac{n}{\log n}\right)^{1/3}$, this lower bound is $\Omega\big(\frac{n}{k^2}\big)$. We remark that the fact that the lower bound still holds even when only a single bit of the answer is to be computed will be crucial for us, as our goal is to reduce the communication complexity problem of

"pointer-following" to computing the Boolean function f_{k+1} in the randomized decision tree model.

Let $\mathcal{A}$ be any (k, q)-round-adaptive (randomized) LDT algorithm computing f_{k+1}. Writing $V_A = \{v_0, \ldots, v_{\frac{n}{2}-1}\}$ and $V_B = \{u_0, \ldots, u_{\frac{n}{2}-1}\}$, fix a bijection between $V \stackrel{\text{def}}{=} V_A \cup V_B$ (of size n) and $\mathbb{F}_n$ mapping v_0 to 1, so that we identify V with $\mathbb{F}_n$. On input (χ_A, χ_B), Alice and Bob implicitly define the element $x \in \mathbb{F}_n^n$ by $x_1 = \chi_A(v_0)$, $x_2 = \chi_A(v_1)$, $\ldots$, $x_{\frac{n}{2}} = \chi_A(v_{\frac{n}{2}-1})$ and $x_{\frac{n}{2}+1} = \chi_B(u_0)$, $x_{\frac{n}{2}+2} = \chi_A(u_1)$, $\ldots$, $x_n = \chi_A(u_{\frac{n}{2}-1})$. From this, we get that $\pi_{k+2}(\chi_A, \chi_B) = g_{k+1}(x)$, recalling that $g_k(x) = g(x, g_{k-1}(x))$ is recursively defined for $k \geq 1$, and $g_0(x) = x_1$. Hence deciding whether $\pi_{k+2}(\chi_A, \chi_B)$ is even is exactly equivalent to computing $f_{k+1}(x)$.

Alice and Bob can then simulate the execution of $\mathcal{A}$ as follows. Without loss of generality, assume it is Alice's turn to speak. To answer a query of the form $\phi_a(x) = \sum_{i=1}^n a_i x_i$, she computes $\sum_{i \in V_A} a_i x_i$ and sends it to Bob; on his side, Bob computes $\sum_{i \in V_B} a_i x_i$, and receiving Alice's message can then recover the value $\phi_a(x)$ and feed it to the algorithm. (In the next round, when sending his side of the (new) queries to Alice, Bob will also send this value $\phi_a(x)$, to make sure that both sides know the answers to all queries so far.) Since all queries of a given adaptive round of $\mathcal{A}$ can be prepared and sent in parallel (costing $O(\log n)$ bits of communication per query), this simulation can be performed in $k + 1$ rounds (as many as $\mathcal{A}$ takes) with communication complexity $O(q \log)$. At the end, whichever of Alice and Bob received the latest message holds the answer (to "is $\pi_{k+1}(\chi_A, \chi_B)$ an even node?"), which by assumption on $\mathcal{A}$ is correct with probability at least 2/3. Alice and Bob then use an extra round of communication to broadcast the answer to the other party, bringing the total number of rounds to $k + 2$.

But by Theorem 9, computing this bit of $\pi_{k+2}(\chi_A, \chi_B)$ with only $k + 2$ rounds of communication (Bob speaking first) requires $\Omega\left(\frac{n}{k^2}\right)$ bits of communication, and so we must have $q = \Omega\left(\frac{n}{k^2 \log n}\right)$. ◀

4.4 Adaptivity Bounded Testers and Decision Trees: There and Back Again

In this section we show how to reduce problems in the adaptivity bounded property testing model to problems in the adaptivity bounded (linear) decision tree model, and vice versa. We begin in Section 4.4.1, by presenting the required preliminaries regarding error-correction codes. Then, in Section 4.4.2, we prove the "transference lemmas" between these models.

4.4.1 Preliminaries: Locally Testable and Decodable Codes

Let $k, n \in \mathbb{N}$. A code over alphabet Σ with distance d is a function $C \colon \Sigma^k \to \Sigma^n$ that maps messages to codewords such that the distance between any two codewords is at least $d = d(n)$. If $d = \Omega(n)$, C is said to have linear distance. If $\Sigma = \{0, 1\}$, we say that C is a binary code. If C is a linear map, we say that it is a linear code. The relative distance of C, denoted by $\delta(C)$, is d/n, and its rate is k/n. When it is clear from the context, we shall sometime abuse notation and refer to the code C as the set of all codewords $\{C(x)\}_{x \in \Sigma^k}$. Following the discussion in the introduction, we define locally testable codes and locally decodable codes as follows.

▶ **Definition 10** (Locally Testable Codes)**.** A code $C \colon \Sigma^k \to \Sigma^n$ is a locally testable code (LTC) if there exists a probabilistic algorithm (tester) T that makes $O(1)$ queries to a purported codeword $w \in \Sigma^n$ and satisfies:

1. Completeness: For any codeword w of C it holds that $\Pr_T[T^w = 1] \geq 2/3$.
2. Strong Soundness: For all $w \in \Sigma^n$,

$$\Pr_T[T^w = 0] \geq \text{poly}\big(\text{dist}(w, C)\big).$$

▶ **Definition 11** (Locally Decodable Codes)**.** A code $C\colon \Sigma^k \to \Sigma^n$ is a locally decodable code (LDC) if there exists a constant $\delta_{\text{radius}} \in (0, \delta(C)/2)$ and a probabilistic algorithm (decoder) D that, given oracle access to $w \in \Sigma^n$ and direct access to index $i \in [k]$, satisfies the following condition: For any $i \in [k]$ and $w \in \Sigma^n$ that is δ_{radius}-close to a codeword $C(x)$ it holds that $\Pr[D^w(i) = x_i] \geq 2/3$. The query complexity of a LDC is the number of queries made by its decoder.

We shall also need the notion of relaxed-LDCs (introduced in [3]). Similarly to LDCs, these codes have decoders that make few queries to an input in attempt to decode a given location in the message. However, unlike LDCs, the relaxed decoders are allowed to output a special symbol that indicates that the decoder detected a corruption in the codeword and is unable to decode this location. Note that the decoder must still avoid errors (with high probability).[6]

▶ **Definition 12** (Relaxed-LDC)**.** A code $C\colon \Sigma^k \to \Sigma^n$ is a relaxed-LDC if there exists a constant $\delta_{\text{radius}} \in (0, \delta(C)/2)$ such that the following holds.

1. (Perfect) Completeness: For any $i \in [k]$ and $x \in \Sigma^k$ it holds that $D^{C(x)}(i) = x_i$.
2. Relaxed Soundness: For any $i \in [k]$ and any $w \in \Sigma^n$ that is δ_{radius}-close to a (unique) codeword $C(x)$, it holds that

$$\Pr[D^w(i) \in \{x_i, \perp\}] \geq 2/3.$$

There are a couple of efficient constructions of codes that are both relaxed-LDCs and LTCs (see [3, 22]). We shall need the construction in [22], which has the best parameters for our setting.[7]

▶ **Theorem 13** (e.g., [22, Theorem 1.1])**.** *For every $k \in \mathbb{N}$, $\alpha > 0$, and finite field $\mathbb{F}$ there exists an $\mathbb{F}$-linear code $C\colon \mathbb{F}^k \to \mathbb{F}^{k^{1+\alpha}}$ with linear distance, which is both a* relaxed-LDC *and a (one-sided error)* LTC *with query complexity* $\text{poly}(1/\varepsilon)$*; furthermore, both testing and (relaxed) decoding procedures are non-adaptive.*

4.4.2 Transference Lemmas

Fix any $\alpha > 0$. Let $C\colon \mathbb{F}_n^n \to \mathbb{F}_n^m$ be a code with constant relative distance $\delta(C) > 0$, with the following properties:

- *linearity*: for all $i \in [m]$, there exists an element $a^{(i)} \in \mathbb{F}_n^n$ such that $C(x)_i = \langle a^{(i)}, x \rangle$ for all $x \in \mathbb{F}_n^n$;

[6] The full definition of relaxed-LDCs, as defined in [3] includes an additional condition on the success rate of the decoder. Namely, for every $w \in \{0,1\}^n$ that is δ_{radius}-close to a codeword $C(x)$, and for at least a ρ fraction of the indices $i \in [k]$, with probability at least $2/3$ the decoder D outputs the i'th bit of x. That is, there exists a set $I_w \subseteq [k]$ of size at least ρk such that for every $i \in I_w$ it holds that $\Pr[D^w(i) = x_i] \geq 2/3$. We omit this condition since it is irrelevant to our application, and remark that every relaxed-LDC that satisfies the first two conditions can also be modified to satisfy the third conditions (see [3, Lemmas 4.9 and 4.10]).

[7] Specifically, the codes in [22] are meaningful for every value of the proximity parameter, whereas the codes in [3] require $\varepsilon > 1/\text{polylog}(k)$.

- *rate*: $m \leq n^{1+\alpha}$;
- *testability*: C is a *strong-LTC* with one-sided error and *non-adaptive* tester;
- *decodability*: C is a *relaxed-LDC*.

We will rely on Theorem 13 for the existence of such codes. Before delving into the details, we briefly explain the reason for each of the points above. The linearity will be crucial to reduce to and from the LDT model: indeed, any coordinate of a codeword corresponds to a fixed linear combination of the coordinates of the message, which corresponds to a single LDT query on that particular linear combination. The rate bound is required since our lower bounds are in terms of the dimension n and upper bounds in terms of the block-lengh m. Ideally, we would like $m = O(n)$, to have a direct correspondence between the LDT and the property testing query complexities; however, this nearly-linear rate is the best known achievable for constant-query LTCs and relaxed-LDCs [22]. The LTC property will be useful to us in the reduction from property testing to DT query complexity (where we will need to first check that our input is close to a codeword, in view of decoding the closest message during the reduction), where the *strong* testability (i.e., rejection with probability proportional to the distance from a valid codeword) will allow us do deal with arbitrarily small values of the proximity parameter. Similarly, we will rely on the (relaxed) LDC property in that same reduction, in order to obtain individual coordinates of the message, given query access to an input close to a codeword.

We proceed to show the framework for reducing property testing to decision tree complexity and vice-versa. For a fixed function $f\colon \mathbb{F}_n^n \to \{0,1\}$, consider the subset $f^{-1}(1) \subseteq \mathbb{F}_n^n$; and define the sets of codewords $\mathcal{C} \stackrel{\text{def}}{=} C(\mathbb{F}_n^n) \subseteq \mathbb{F}_n^m$, $\mathcal{C}_f \stackrel{\text{def}}{=} C(f^{-1}(1)) = \{\, C(x) : x \in \mathbb{F}_n^n,\ f(x) = 1 \,\} \subseteq \mathcal{C}$.

Consider now testing the property $\mathcal{C}_f$: we will reduce the LDT computation of f to the testing of $\mathcal{C}_f$. Specifically, we prove the following.

▶ **Lemma 14** (LDT $\rightsquigarrow$ PT Reduction Lemma)**.** *Fix any $f\colon \mathbb{F}_n^n \to \{0,1\}$. If there exists an (k,q)-round-adaptive tester for $\mathcal{C}_f$, then there is an (k,q)-round-adaptive LDT algorithm for f.*

Proof. Suppose there exists a (k,q)-round-adaptive tester $\mathcal{T}$ for $\mathcal{C}_f$. On input $x \in \mathbb{F}_n^n$, we emulate the invocation of $\mathcal{T}$, with respect to proximity parameter $\varepsilon = \delta(C)$, on the encoded input $y \stackrel{\text{def}}{=} C(x) \in \mathbb{F}_n^m$ and output 1 if and only if $\mathcal{T}$ returns accept. To see why this is correct, observe that by definition, if $f(x) = 1$ then $y \in \mathcal{C}_f$. However, if $f(x) = 0$, then for any $y' \in \mathcal{C}_f$ such that $y' = C(x)$ we must have $\mathrm{dist}(y, y') > \varepsilon$, by the distance of our code.

It remains to show that this simulation can be achieved efficiently, as claimed. To do so, we will rely on the fact that C is a linear code: whenever $\mathcal{T}$ queries y_i, we can compute the element $a^{(i)} \in \mathbb{F}_n^n$ (which only depends on C, and not on x), and perform the LDT query $\langle a^{(i)}, x \rangle$. The simulation clearly preserves the number of adaptive rounds as well, concluding the proof. ◀

In our next lemma, we give a partial converse relating property testing and decision tree complexity, with some logarithmic overhead in the resulting query complexity.

▶ **Lemma 15** (PT $\rightsquigarrow$ DT Reduction Lemma)**.** *Fix any $f\colon \mathbb{F}_n^n \to \{0,1\}$. If there exists an (k,q)-round-adaptive (randomized) DT algorithm for f, then there is a $(k, O(q \log q) + \mathrm{poly}(1/\varepsilon))$-round-adaptive tester for $\mathcal{C}_f$. (Moreover, if the DT algorithm is always correct, then this tester is one-sided.)*

Proof. Fix $k \geq 0$, and suppose there exists such a (k, q)-round-adaptive DT algorithm $\mathcal{A}$ for f. On input $y \in \mathbb{F}_n^m$ and proximity parameter $\varepsilon \in (0, 1]$, we would like to decode y to a message $x \in \mathbb{F}_n^n$ and invoke the algorithm on x to determine if $f(x) = 1$; more precisely, we wish to invoke the DT algorithm while simulating each query to x by locally decoding y using $O(1)$ queries. The issue, however, is that the success of the local decodable is only guaranteed for inputs that are sufficiently close to a valid codeword, and we have no such guarantee on y *a priori*. However, recalling that C is a strong-LTC, we can handle this as follows. Letting $\delta_{\mathsf{radius}} > 0$ be the decodability radius of the relaxed-LDC C, we set $\delta^* \stackrel{\text{def}}{=} \min(\delta_{\mathsf{radius}}, \varepsilon)$.

1. Run independently $O(\text{poly}(1/\delta^*))$ times the local tester for the strong-LTC C on y, and output reject if any of these rejected. Since every invocation of the local tester makes $O(1)$ queries to y, this has query complexity $O(\text{poly}(1/\delta^*)) = O(\text{poly}(1/\varepsilon))$; and if $\text{dist}(y, \mathcal{C}) > \delta^*$ then this step outputs reject with probability at least $9/10$.
2. Invoke $\mathcal{A}$ on the message $x \stackrel{\text{def}}{=} \text{argmin}\{\, \text{dist}(C(x), y) : \; x \in \mathbb{F}_n^n \,\}$, answering each query x_i by calling the local decoder for the relaxed-LDC C. This is done so that the decoder is correct with probability at least $1/(10q)$, by standard repetition (taking the plurality value); with the subtlety that we output reject immediately whenever the decoder returns $\perp$. Since each query can be simulated by $O(\log(q))$ queries (repeating the $O(1)$ queries of the decoder $O(\log q)$ times), this step has query complexity $O(q \log q)$; and at the end, we output accept if, and only if, $\mathcal{A}$ returns the value 1 for $f(x)$.

Importantly, Step 1 can be run in parallel to Step 2, and in particular can be executed during the first "batch" of queries $\mathcal{A}$ makes. This guarantees that the whole simulation above uses the same number of adaptive rounds as $\mathcal{A}$, as claimed. It remains to argue correctness.

Completeness. Assume $y \in \mathcal{C}_f$. In particular, y is a codeword of C, and the (one-sided) local tester returns accept with probability one in 1. Then, since by definition there is a unique $x \in \mathbb{F}_n^n$ such that $C(x) = y$, the local decoder of Step 2 will correctly output the correct answer for each query with probability 1, and therefore $\mathcal{A}$ will correctly output $f(x)$ with probability $2/3$ – so that the tester returns accept with probability at least $2/3$ overall. (Moreover, if the DT algorithm $\mathcal{A}$ always correctly compute f, then the tester returns accept with probability one.)

Soundness. Assume $\text{dist}(y, \mathcal{C}_f) > \varepsilon$. If $\text{dist}(y, \mathcal{C}) > \delta^*$, then the local tester returns reject with probability at least $9/10$ in Step 1. Therefore, we can continue assuming that $\text{dist}(y, \mathcal{C}) \leq \delta^*$, which satisfies the precondition of the relaxed-LDC decoder in Step 2. By a union bound over all q queries, with probability at least $9/10$ we have that the decodings performed in Step 2 are all correct; in which case we answer the queries of the algorithm according to $x \stackrel{\text{def}}{=} \text{argmin}\{\, \text{dist}(C(x), y) : \; x \in \mathbb{F}_n^n \,\}$ (or possibly answered by $\perp$, in which case the tester immediately outputs reject and we are done). Since $\text{dist}(y, C(x)) \leq \delta^* \leq \varepsilon$, we must have $C(x) \notin \mathcal{C}_f$, which implies that $\mathcal{A}$ correctly returns $f(x) = 0$ with probability at least $2/3$, in which case the tester outputs reject. Overall, this happens with probability at least $9/10 \cdot 9/10 \cdot 2/3 = 27/50$.

Thus, in both cases the tester is correct with probability at least $27/50$; repeating a constant number of times (as explained in the remark of page 27:6) and taking the majority vote allows us to amplify the probability of success to $2/3$. ◀

5 An Adaptivity Hierarchy with respect to a Natural Property

In this section we show a *natural* property of graphs for which, broadly speaking, more adaptivity implies more power. More specifically, we prove the following adaptivity hierarchy theorem with respect to the property of k-cycle freeness in the bounded-degree graph model (see definitions in Section 5.1).

▶ **Theorem 16.** *Let $k \in \mathbb{N}$ be a constant. Then,*

1. *there exists a $(k, O(1/\varepsilon))$-round-adaptive (one-sided) tester for $(2k+1)$-cycle freeness in the bounded-degree graph model; yet*
2. *any $(k-1, q)$-round-adaptive (two-sided) tester for $(2k+1)$-cycle freeness in the bounded-degree graph model must satisfy $q = \Omega(\sqrt{n})$.*

We stress that although Theorem 5 establishes an adaptivity hierarchy with stronger separations, the merit of Theorem 16 is in showing that an adaptivity hierarchy also holds for a *natural* well-studied property. We further observe that the choice of the bounded-degree graph model is not insignificant: one cannot hope to establish such a striking gap in other settings such as the dense graph model or in the Boolean function testing setting. Indeed, as discussed in Section 1.2 it is well-known that in these two models, any adaptive tester can be made (fully) non-adaptive at the price of only a quadratic and exponential blowup in the query complexity, respectively(see [2, 25] for the former; the latter is folklore). We remark that in Section 6.1 we discuss emulating testers with k rounds of adaptivity by testers with $k' < k$ rounds.

5.1 Cycle Freeness in the Bounded Degree Graph Model

In the subsection we provide the necessary definitions and establish a basic upper bound on the complexity of k-adaptive testing of cycle freeness in the bounded degree graph model. We begin with a definition of the model.

Let $G = (V, E)$ be a graph with constant degree bound $d < |V|$, represented by its adjacency list; that is, represented by a function $g : V \times d \to V$ such that $g(v, i) = u \in V$ if u is the ith neighbor of v and $g(v, i) = 0$ if v has fewer than i neighbors. A bounded degree graph property $\mathcal{P}$ is a subset of graphs (represented by their adjacency list) that is closed under isomorphism; that is, for every permutation π it holds that $G \in \mathcal{P}$ if and only if $G \in \pi(G)$. The distance of graph G from property $\mathcal{P}$ is the minimal fraction of entries in g one has to change to reach an element of $\mathcal{P}$.

We extend the definition of functional round-adaptive testing algorithms to the bounded degree graph model in the natural way.

▶ **Definition 17** (Round-Adaptive Testing in the Bounded Degree Graph Model)**.** Let $G = (V, E)$ be a graph with constant degree bound $d < |V|$, represented by its adjacency list $g \colon V \times d \to V$, and let $k, q \leq n$. A randomized algorithm is said to be a (k, q)-*round-adaptive* tester for a (bounded degree) graph property $\mathcal{P}$, if, on proximity parameter $\varepsilon \in (0, 1]$ and granted query access to g, the following holds.

1. Query Generation: The algorithm proceeds in $k+1$ rounds, such that at round $\ell \geq 0$, it produces a set of queries $Q_\ell \stackrel{\text{def}}{=} \{x^{(\ell),1}, \ldots, x^{(\ell),|Q_\ell|}\} \subseteq [n]$ (possibly empty), based on its own internal randomness and the answers to the previous sets of queries $Q_0, \ldots, Q_{\ell-1}$, and receives $f(Q_\ell) = \{g(x^{(\ell),1}), \ldots, g(x^{(\ell),|Q_\ell|})\}$;
2. Completeness: If $G \in \mathcal{P}$, then the algorithm outputs accept with probability at least $2/3$;
3. Soundness: If $\mathrm{dist}(G, \mathcal{P}) > \varepsilon$, then the algorithm outputs reject with probability at least $2/3$.

The *query complexity* q of the tester is the total number of queries made to f, i.e., $q = \sum_{\ell=0}^{k} |Q_\ell|$. If the algorithm returns accept with probability one whenever $f \in \mathcal{P}$, it is said to have *one-sided* error (otherwise, it has *two-sided* error). As before, we will sometimes refer to a tester with respect to proximity parameter ε as an ε-tester.

Next, we define the (bounded degree) graph property of k-cycle freeness.

▶ **Definition 18** (Cycle Freeness)**.** Let $k \in \mathbb{N}$. A graph $G = (V, E)$ is said to be k-cycle free if it does not contain any cycle of length less or equal to k; that is, if for every $t \leq k$ and $v_1, \ldots, v_t \in V$ either $(v_t, v_1) \notin E$ or there exists $i \in [t-1]$ such that $(v_i, v_{i+1}) \notin E$.

Finally, we make the following observation, which roughly speaking implies that when surpassing a certain threshold of round adaptivity, testing cycle freeness in the bounded degree graph model becomes "easy."[8]

▶ **Observation 19.** *For every $k \in \mathbb{N}$ there exists a (k, q)-round-adaptive testing algorithm for $(2k+1)$-cycle freeness and $(2k+2)$-cycle freeness in the bounded-degree graph model with query complexity $q = O(d^{k+1}/\varepsilon)$.*

Proof. The algorithm explores the graph in the most natural way: starting from $O(1/\varepsilon)$ "source vertices" selected uniformly at random, it adaptively explore their neighborhoods by querying at each round the neighbors of the previously reached vertices, in a breadth-first-search fashion. If any $(2k+1)$-cycle (resp. $(2k+2)$-cycle) is detected, the algorithm rejects, and accepts otherwise. (Clearly, this tester is one-sided.) It is easy to see that if any of the source vertices belongs to a $(2k+1)$- or $(2k+2)$-cycle, then this bounded-depth BFS will detect it; thus, we only need to argue that if the graph is ε-far from cycle freeness, with constant probability, one of the source vertices will participate in such a cycle. But this is the case, as any such graph must have at least εn vertices participating in a cycle (indeed, otherwise one could "correct" the graph by removing fewer than εdn vertices, contradicting the distance).

Finally, for each source vertex, after k rounds of adaptivity the number of nodes visited is at most $O(d^{k+1})$, hence the claimed query complexity. ◀

5.2 Lower Bounds for Round-Adaptive Testers

In this subsection, we prove the following lemma, which roughly speaking shows that testing $(2k+3)$-cycle freeness is hard for k-round-adaptive testing algorithms.

▶ **Lemma 20.** *Let $k \in \mathbb{N}$ be constant. Then, any (k, q)-round-adaptive testing algorithm for $(2k+3)$-cycle freeness in the bounded-degree graph model must satisfy $q = \Omega(\sqrt{n})$.*

In stark contrast, recall that Observation 19 shows that testing $(2k+2)$-cycle freeness is easy for k-round-adaptive testing algorithms. Indeed, the proof of Theorem 16 follows by combining Observation 19 and Lemma 20 together.

Proof. Proof of Lemma 20 We will show a distribution of $(2k+3)$-cycle free graphs, denoted $\mathcal{Y}$, and a distribution of graphs that are "far" from being $(2k+3)$-cycle free, denoted $\mathcal{N}$, and prove that no (k, q)-round-adaptive testing algorithm can distinguish, with high probability, between $\mathcal{Y}$ and $\mathcal{N}$. Loosely speaking, $\mathcal{Y}$ consists of all graphs whose vertices are covered via

[8] This is a specific case of a more general algorithm for testing subgraph freeness; see e.g. [20, Section 9.2.1].

disjoint $(2k+4)$-cycles, and $\mathcal{N}$ consists of all graphs whose vertices are covered via disjoint $(2k+3)$-cycles.

More accurately, denote by $\mathcal{P}_{t,n,d}$ the subset of n-node graphs with maximum degree at most d that are t-cycle-free. Let $\Sigma_{t,s}$ be the 2-regular graph on st vertices made of s disjoint t-cycles, namely $(v_1,\dots,v_t)$, $(v_{t+1},\dots,v_{2t})$, $(v_{(s-1)t+1},\dots,v_{st})$. Denote also by Is_r the independent set on r vertices. For two graphs G, G' on respectively m and m' vertices and with e and e' edges, we write $G \sqcup G'$ for the graph on $m+m'$ vertices and with $e+e'$ edges obtained by concatenating disjoint copies of G, G'.

For $k = O(1)$, we let $\ell \stackrel{\text{def}}{=} \left\lfloor \frac{n}{(2k+4)} \right\rfloor$, $\ell' \stackrel{\text{def}}{=} \left\lfloor \frac{n}{(2k+3)} \right\rfloor$, and define the two distributions over n-node graphs $\mathcal{Y}$ and $\mathcal{N}$ as follows.

- $\mathcal{Y}$ is the uniform distribution over all isomorphic copies of $G_k^{\mathsf{yes}} \stackrel{\text{def}}{=} \Sigma_{(2k+4),\ell} \sqcup \mathrm{Is}_{n-(2k+4)\ell}$;
- $\mathcal{N}$ is the uniform distribution over all isomorphic copies of $G_k^{\mathsf{no}} \stackrel{\text{def}}{=} \Sigma_{(2k+3),\ell'} \sqcup \mathrm{Is}_{n-(2k+3)\ell'}$.

The next claim establishes that indeed $\mathcal{Y}$ consists of yes-instances, whereas $\mathcal{N}$ consists of no-instances.

▶ **Claim 21.** *$\mathcal{Y}$ is supported on $\mathcal{P}_{(2k+3),n,d}$, while every graph in the support of $\mathcal{N}$ is $\Omega(1)$-far from $\mathcal{P}_{(2k+3),n,d}$.*

Proof. The first part is obvious, as the only cycles in G_k^{yes} are $(2k+4)$-cycles. As for the second, it immediately follows from observing that G_k^{no} contains ℓ' disjoint $(2k+3)$-cycles, and thus at least ℓ' edges have to be removed to make it $(2k+3)$-cycle free. Thus, $\mathrm{dist}\big(G_k^{\mathsf{no}}, \mathcal{P}_{(2k+3),n,d}\big) \geq \frac{\ell'}{dn/2} = \Omega\big(\frac{1}{dk}\big) = \Omega_d(1)$. ◀

Let $\mathcal{T}$ be a deterministic testing algorithm with k rounds of adaptivity and query complexity $q' = o(\sqrt{n})$. The following lemma concludes the proof of Lemma 20 by showing that $\mathcal{T}$ cannot distinguish, with high probability, between graphs in $\mathcal{Y}$ and graphs in $\mathcal{N}$. Denote $\mathcal{T}$'s (disjoint) query sets, per round, by $Q_0,\dots,Q_k \subseteq V$, where a query is a vertex v. Denote the corresponding sets of answers by $A_0,\dots,A_k$, where the answer to a query v consists of the labels of all neighbors of v (i.e., either two or zero vertices). Since $k = O(1)$, without loss of generality, we can assume (by padding) that all query sets have the same size $q \stackrel{\text{def}}{=} |Q_i| = \frac{q'}{k+1} = \Theta(q')$ for every $i \in \{0,\dots,k\}$. Moreover, we can also assume that no vertex is queried twice, i.e. that all Q_i's are disjoint.

▶ **Lemma 22.** $\left|\Pr_{G\sim\mathcal{Y}}\left[\mathcal{T}^G \text{ accepts}\right] - \Pr_{G\sim\mathcal{N}}\left[\mathcal{T}^G \text{ accepts}\right]\right| \leq \frac{1}{10}$.

Proof. For $j \in \{0,\dots,k\}$, define by Y_j and N_j the distribution of $(A_0,\dots,A_j)$ when $G \sim \mathcal{Y}$ and when $G \sim \mathcal{N}$, respectively. We shall prove that $\mathrm{d_{TV}}(Y_k, N_k) \leq \frac{1}{10}$, which by the data processing inequality will imply the claim of Lemma 22.

The high-level idea is that in each round, the tester can either query "fresh" vertices, of which it has no prior information, or query the boundaries (i.e., the direct neighbors) of previously queried vertices. Then, loosely speaking we can argue that, on the one hand, if the total number of queries is $o(\sqrt{n})$, then both for graphs in $\mathcal{Y}$ and $\mathcal{N}$ all queries of "fresh" vertices (obtained during all rounds) with high probability would only fall into previously unattained disjoint cycles, in which case the answer would be a uniform sequence of "fresh" labels. On the other hand, the local view obtained by querying the boundary, using at most k rounds of adaptive queries, of each vertex previously obtained via a "fresh" query (which by the above lies in a cycle wherein the tester has no information of the labels of the other vertices participating in this cycle) is isomorphic to the tail graph over fresh labels, both for instances taken from $\mathcal{Y}$ and $\mathcal{N}$ (that is, we do not have enough adaptive queries to observe a full cycle). The foregoing intuition is formalized below.

For $i \in \{0, \ldots, k\}$, define

$$S_i^{\mathrm{f}} \stackrel{\mathrm{def}}{=} Q_i \setminus \cup_{j=0}^{i-1} A_j$$
$$S_i^{\mathrm{b}} \stackrel{\mathrm{def}}{=} Q_i \cap \cup_{j=0}^{i-1} A_j$$

to be, respectively, the set of "entirely fresh" nodes queried at round i (that is, nodes that are not neighbors of any previously queried node), and the set of "boundary nodes" (which are the not-yet-queried nodes neighbors of a previously queried node).

First, we bound the probability that any of the q' queries made "hits" the set of disconnected nodes:

▶ **Claim 23.** *Let $E_1(G)$ denote the event that $\mathcal{T}$ queries an isolated vertex of G, that is $E_1(G) \stackrel{\mathrm{def}}{=} \{\exists i, v \text{ s.t. } v \in Q_i,\ \deg(v) = 0\}$. Then $\Pr_{G\sim\mathcal{Y}}[E_1(G)], \Pr_{G\sim\mathcal{N}}[E_1(G)] = o(1)$.*

Proof. This follows by induction: at step i, conditioned on no isolated node having been queried yet, the algorithm has degree information about $\left|\cup_{j=0}^{i-1} Q_j \bigcup \cup_{j=0}^{i-1} A_j\right| \leq \sum_{j=0}^{i-1} |Q_j| + \sum_{j=0}^{i-1} |A_j| \leq 3q \cdot i$ nodes, so there remain at least $n - 3kq$ nodes on which the algorithm has no degree information at all. Among these, there are $n - (2k+4)\ell \leq (2k+4)$ (or $n - (2k+3)\ell' \leq (2k+3)$, in the no-case) isolated nodes. By symmetry, this means that in the new batch of q queries, the algorithm will query one of these isolated nodes with probability at most $1 - \left(1 - \frac{(2k+4)}{n-3kq-(2k+4)}\right)^q = 1 - \left(1 - \frac{O(1)}{n}\right)^q = O\left(\frac{q}{n}\right) = o(1)$. Therefore, overall there will be an isolated node queried with probability at most $k \cdot o(1) = o(1)$. ◀

Next, we argue that at each step, with overwhelming probability all the "fresh nodes" queried fall in distinct cycles, which have not been attained yet.

▶ **Claim 24.** *Let $E_2(G)$ denote the event that at some round i, one of the queries in S_i^{f} belongs to the same cycle (either a $(2k+4)$- or a $(2k+3)$-cycle, depending on whether the graph is drawn from $\mathcal{Y}$ or $\mathcal{N}$) as one of the previous queries $\bigcup_{j=0}^{i-1} Q_j$. Then $\Pr_{G\sim\mathcal{Y}}[E_2(G)], \Pr_{G\sim\mathcal{N}}[E_2(G)] = o(1)$.*

Proof. We will show that $\Pr_{G\sim\mathcal{Y}}[E_2(G)] = o(1)$; the no-case is similar. For $i \in \{1, \ldots, k\}$, let $E_2^{(i)}(G)$ denote the event that at some round i, one of the queries in S_i^{f} belongs to the same cycle as a previous query, so that $E_2(G) = \bigcup_{i=1}^{k} E_2^{(i)}(G)$.

Note that since $|\bigcup_{j=0}^{i-1} Q_j| = iq$, we have $|\bigcup_{j=0}^{i-1} A_j| \leq 2iq$ (and the number of distinct cycles reached is at most $|\bigcup_{j=0}^{i-1} Q_j|$). Therefore, at round i each of the at most q distinct queries in S_i^{f} falls independently in a previously visited cycle with probability upper bounded by

$$\frac{iq \cdot (2k+4)}{n - 3iq} \leq \frac{kq \cdot (2k+4)}{n - 3kq} \leq \frac{2k^2 q}{n}$$

recalling that $q = o(n)$ and $k = O(1)$. A union bound over all at most q queries of S_i^{f}, and then over the k rounds then shows that $\Pr_{G\sim\mathcal{Y}}[E_2(G)] \leq \frac{2k^3 q^2}{n} = o(1)$ (since $q = o(\sqrt{n})$). ◀

To conclude the proof, note that by the above, with probability $1 - o(1)$ neither E_1 nor E_2 occurs; that is, none of the isolated vertices was queried, and all the "fresh" queries (during all rounds) fell in previously unattained distinct cycles. In this case, at each round of adaptivity the algorithm can at most discover two new nodes out of every cycle it reached before (by including the one or two end nodes of the current "discovered portion" into S_i^{b}). Therefore, on any cycle ever reached, the (k, q)-round-adaptive testing algorithm can observe

at most $2k+2$ nodes (which then form a consecutive path). We show that this implies that the algorithm cannot distinguish between a no-instance and a yes-instance, as loosely speaking, in both cases its local view is of a tail graph over uniformly distributed fresh labels, and so it is unable to determine whether it belongs to a cycle of length $2k+3$ or $2k+4$.

To make the argument more precise, we will actually show a stronger statement; namely, we show that, conditioning on neither E_1 nor E_2 occuring, a simulator with no access to the graph can answer the queries of the testing algorithm in a way that is indistinguishable from the tuple of answers obtained from querying a graph distributed according to either $\mathcal{Y}$ or $\mathcal{N}$. This simulator operates as follows: at round i,

1. Order (arbitrarily) all the nodes of Q_i: $v_1, \ldots, v_q$, and initialize the set of available-to-sample nodes $U \leftarrow V \setminus \left(Q_i \cup \bigcup_{j=0}^{i-1} Q_j \cup \bigcup_{j=0}^{i-1} A_j\right)$.
2. Do sequentially the following, for $s = 1 \ldots q$:
   - if $v_s \in S_i^{\mathrm{f}}$ (fresh node: no previous neighbors known), pick uniformly at random two distinct nodes u, u' in U_s and return them as answers (i.e., declare them as neighbors of v_s);
   - otherwise, $v_s \in S_i^{\mathrm{b}}$ (boundary node: exactly one already known neighbor, call it u): pick uniformly at random one other node u' in U_s, and return (u, u') as answers;
   - update U by removing u, u': $U \leftarrow U \setminus \{u, u'\}$

It is straightforward to verify that, since we conditioned on $\overline{E_1}$ and $\overline{E_2}$, this simulates exactly the same distribution over nodes (over the choice of G); since this is the same both for $\mathcal{Y}$ and $\mathcal{N}$, we get that $\mathrm{d_{TV}}\big((Y_k \mid \overline{E_1 \cup E_2})), (N_k \mid \overline{E_1 \cup E_2}))\big) = 0$, which combined with Claim 23 and Claim 24 finishes the proof. ◀

This concludes the proof of Lemma 20. ◀

6 Some Miscellaneous Remarks

In this section we discuss adaptivity round reductions, as well as a connection to communication complexity, and the relation between round and tail adaptivity. Specifically, in Section 6.1 we show how to simulate k rounds of adaptivity via $k-1$ rounds (at the cost of an increase in query complexity). In Section 6.2 we extend the communication complexity methodology for proving property testing lower bounds [9] to k-round adaptive testers, then sketch an alternative proof of item (2) of Theorem 5 using it; and show how it can also be leveraged to prove a hierarchy of lower bounds on the power of k-adaptive testers for a fundamental class of Boolean functions. Finally, in Section 6.3 we show a separation between the power of round-adaptive and tail-adaptive testers.

6.1 On Simulating k Rounds With Fewer

As mentioned in the beginning of Section 5, in the Boolean setting any adaptive property testing algorithm can be simulated non-adaptively with only an exponential blowup in the query complexity. Phrased differently, this implies that any property of Boolean functions which admits a (k, q)-round-adaptive tester also has a $(0, 2^q - 1)$-round-adaptive tester.

This begs the following more general question: let $\mathcal{P} = \bigcup_n \mathcal{P}_n$ be a property of Boolean functions, such that there exists a (k, q)-round-adaptive tester for $\mathcal{P}$. For $\ell < k$, what upper bound can we obtain on the query complexity q' of the best (ℓ, q')-round-adaptive tester for $\mathcal{P}$?

Denoting by q_ℓ this query complexity, the above discussion immediately implies:

▶ **Fact 25.** *For any $0 \leq \ell \leq k$, one has $q_k \leq q_\ell \leq 2^{q_k} - 1$.*

In what follows, we provide a example of a more fine-grained version of this fact, in the case when $\ell = k - 1$ (that is, one wishes to reduce the number of rounds of adaptivity by one).

▶ **Proposition 26.** *For any $0 < k$, one has $q_k \leq q_{k-1} \leq q_k(1 + 2^{\frac{q_k}{k}})$.*

Proof. Let $\mathcal{T}_k$ be a (k, q)-round-adaptive tester for $\mathcal{P}$, which can be viewed as a distribution over deterministic algorithms. Thus, it is sufficient to explain how to simulate any deterministic algorithm with k rounds of adaptivity by one with ℓ rounds. Fix such a (k, q)-round deterministic algorithm: this can be seen equivalently as a depth-$(k + 1)$ binary tree, where each internal node v is labeled by the set of queries Q_v made at that stage, and the leaves are either accept or reject. By assumption, we have that on each path $(v_0, v_1, \ldots, v_k, v^*)$ from the root to a leaf, $\sum_{j=0}^{k} \left|Q_{v_j}\right| \leq q$; moreover, one can assume without loss of generality that this is an equality.

The idea is then to contract, on any path, two consecutive nodes as follows: instead of querying Q_{v_j}, receiving the answers, and then querying the (adaptively chosen) set $Q_{v_{j+1}}$, one can idea query simultaneously Q_{v_j} and the union of all possible sets $Q_{v_{j+1}}$: since the latter depends only on the previous queries, and the only unknown answers are those to the queries in Q_{v_j}, there are at most $2^{|Q_{v_j}|}$ possibilities for $Q_{v_{j+1}}$. As clearly no matter what $Q_{v_{j+1}}$ would be, its size is at most q, the set $Q'_i = Q_{v_j} \cup \bigcup_{Q:\ \text{possible } Q_{v_{j+1}}} Q$ queried has size at most $\left|Q_{v_j}\right| + q2^{|Q_{v_j}|}$. Thus, by contradicting the two rounds i and $i + 1$, one incurs an additional number of queries upper bounded by $q2^{|Q_{v_j}|} - \left|Q_{v_{j+1}}\right| \leq q2^{|Q_{v_j}|}$

By an averaging argument, since on every such path we have $\sum_{j=0}^{k} \left|Q_{v_j}\right| = q$, there must exist an index j^* such that $\left|Q_{v_{j^*}}\right| \leq \frac{q}{k+1}$. Since we would like to "contract" rounds j^* and $j^* + 1$ into a single round, we additionally want to ensure $j^* < k$. But similarly, as $\sum_{j=0}^{k-1} \left|Q_{v_j}\right| \leq q$ there exists i^* such that $|Q_{v_{i^*}}| \leq \frac{q}{k}$. We then get an index $i^* < k$ (which depends on the path taken down the tree) to which we can apply the above transformation. That is, whenever the deterministic algorithm is executed it will reach an index $i^* < k$ where it should make $|Q_{v_{i^*}}| \leq \frac{q}{k}$ queries. At that point, it makes instead these queries, along with all queries this should have triggered at the next round, and thus is able to skip round $i^* + 1$ at the price of an additional (at most) $q2^{\frac{q}{k}}$ queries. ◀

▶ Remark. Note that in the above proof, while one can assume without loss of generality that the algorithm always makes exactly q queries, one *cannot* however assume that for any two such paths $(v_0, v_1, \ldots, v_k, v^*)$ and $(u_0, u_1, \ldots, u_k, u^*)$, $\left|Q_{v_j}\right| = \left|Q_{u_j}\right|$ for all $0 \leq j \leq k$. That is, the number of queries made in round j may not be the same depending on the path followed down by the algorithm, but instead depend adaptively on the previous queries made.

The above remark shows the difficulty in extending the proof of Proposition 26 further than a single round. If one is willing to assume that the number of queries at each round is non-adaptive, it becomes possible to obtain a more general statement for $0 \leq \ell < k$; however, it is unclear how to proceed without this extra assumption, leading to the following question:

▶ **Open Problem 3.** *Can one obtain a general round-reduction upper bound for $0 \leq \ell < k$ of the form $q_\ell \leq \phi(q_k, \ell, k)$, improving on Fact 25 for $\ell > 0$?*

6.2 On the Connection with Communication Complexity

As exemplified in the proof of Lemma 8, there exists a striking parallel between the notion of k-round-adaptive testing algorithms, and that of k-round protocols in communication

complexity. In this section, we make this parallel rigorous, and give a blackbox reduction between the two that one can leverage to establish lower bounds on k-round-adaptive testing.

In more detail, we build on the communication complexity methodology for proving property testing lower bounds due to [9] (more precisely, to the general formulation of this methodology as laid out in [19]). Although the results stated there hold for non-adaptive lower bounds (in the case of one-way communication or simultaneous message passing) or fully adaptive lower bounds in property testing (in the case of two-way communication), it is easy to obtain their counterpart for k-round-adaptive, given in Theorem 27 below. But first, we need to recall some notations.

In what follows, for a property $\mathcal{P}$, integer k, and parameters $\varepsilon, \delta \in [0,1]$, we write $Q_\delta^{(k)}(\varepsilon, \mathcal{P})$ for the minimum query complexity of any k-round-adaptive tester for $\mathcal{P}$ with error probability δ and distance parameter ε. Given a communication complexity predicate F, we let $\mathrm{CC}_\delta^{(k)}(F)$, $\overrightarrow{\mathrm{CC}}_\delta(F)$, and $\overleftarrow{\mathrm{CC}}_\delta(F)$ denote respectively the minimum communication complexity of a public-coin protocol for F with error δ in (i) k-rounds, (ii) one-way from Alice to Bob, and (iii) one-way from Bob to Alice, respectively (note that the case $\delta = 0$ then corresponds to protocols with perfect completeness).

▶ **Theorem 27.** *Let $\Psi = (P, S)$ be a promise problem such that $P, S \subseteq \{0,1\}^{2n}$, $\mathcal{P} \subseteq \{0,1\}^\ell$ be a property, and $\varepsilon, \delta > 0$. Suppose the mapping $F \colon \{0,1\}^{2n} \to \{0,1\}^\ell$ satisfies the following two conditions:*

1. *for every $(x,y) \in P \cap S$, it holds that $F(x,y) \in \mathcal{P}$;*
2. *for every $(x,y) \in P \setminus S$, it holds that $F(x,y)$ is ε-far from $\mathcal{P}$.*

Then $Q_\delta^{(k)}(\varepsilon, \mathcal{P}) \geq \frac{1}{B+1} \mathrm{CC}_{2\delta}^{(k+2)}(\Psi)$, where $B \stackrel{\text{def}}{=} \max_{i \in [\ell]} \max(\overrightarrow{\mathrm{CC}}_{\frac{\delta}{n}}(F_i), \overleftarrow{\mathrm{CC}}_{\frac{\delta}{n}}(F_i))$ (and $F_i(x,y)$ is the i'th bit of $F(x,y)$). Moreover, if $B' \stackrel{\text{def}}{=} \max_{i \in [\ell]} \max(\overrightarrow{\mathrm{CC}}_0(F_i), \overleftarrow{\mathrm{CC}}_0(F_i))$, then $Q_\delta^{(k)}(\varepsilon, \mathcal{P}) \geq \frac{1}{B'+1} \mathrm{CC}_\delta^{(k+2)}(\Psi)$.

Proof. The proof will be identical to that of [19, Theorem 3.1], where we only need to check that Alice and Bob can each simulate the execution of the property testing algorithm (using their public random coins), answering the queries made to $F(x,y)$ while preserving the number of rounds. Running the testing algorithm, Alice first sends the bits allowing Bob to compute the answers to the first q_0 queries, using her input x and the one-way protocols for the relevant F_i's. Bob then answers with the q_0 bits corresponding to the answers he computed, as well as the bits allowing Alice to compute the answers to the next q_1 queries made by the tester, using now his input y and the one-way protocols for the relevant F_i's. They do so for $k+1$ rounds of communication in total, until the last player to receive a message gets from the other player both the answers to the queries in Q_{k-1} as well as the bits needed to compute (given their own input) the answers to the last q_k queries. At that point, it only remains to use a last round of communication (the $(k+2)$'nd) to communicate to the other player the answers to these last q_k queries, so that both Alice and Bob can finish running their copy of the testing algorithm and know the answer.

Note that the number of bits communicated at round $1 \leq i \leq k+2$ is by definition of B (resp. B') at most $B \cdot q_{i-1} + q_{i-2}$ (resp. $B' \cdot q_{i-1} + q_{i-2}$), so that at most $(B+1)q$ (resp. $(B'+1)q$) bits are communicated in total. This concludes the proof. ◀

To illustrate the above methodology, we show how it can be leveraged to prove a hierarchy of lower bounds on the power of k-adaptive testers for testing a very fundamental class of Boolean functions, that of m-linear functions.[9]

[9] We observe that establishing the upper bound counterpart to this result would provide an answer to

▶ **Proposition 28.** *Let* $\mathsf{PAR}^n_s \subseteq 2^{2^n}$ *denote the class of parities of size* s *(over* n *variables), and fix* $m \stackrel{\text{def}}{=} \frac{\sqrt{n}}{2}$*. Then, for any* $0 \le k \le \log^* m - 2$*, any* (k,q)*-round-adaptive tester for* PAR^n_{2m} *must satisfy* $q = \Omega\Big(m \log^{(k+2)} m\Big)$*.*

Proof. We will rely on a result of Sağlam and Tardos [37], which implies the following (tight) lower bound on the communication complexity of sparse set-disjointness (DISJ^n_m, where both inputs $x, y \in \{0,1\}^n$ are promised to have Hamming weight m):

▶ **Theorem** (Corollary of [37, Theorem 4])**.** *For any* $1 \le k \le \log^* m$*, any* k*-round probabilistic protocol for* $\mathsf{DISJ}^{4m^2}_m$ *with error probability at most* $1/3$ *must have communication* $\Omega\Big(m \log^{(k)} m\Big)$*.*

It then suffices to provide a reduction from $\mathsf{DISJ}^{4m^2}_m$ to testing $\mathsf{PAR}^{4m^2}_{2m}$. We follow the known reduction, as can be found in [9, 14]. Namely, on input $x \in \{0,1\}^n$ (resp. $y \in \{0,1\}^n$), Alice (resp. Bob) forms the parity function χ_x (resp. χ_y). As $|x \oplus y| = |x| + |y| - 2\,|x \cap y| = 2m - 2\,|x \cap y|$, the function $\chi_{x \oplus y}$ is a $2(m - |x \cap y|)$-parity. Moreover, as for any $z \in \{0,1\}^n$ we have $\chi_{x\oplus y}(z) = \chi_x(z) \oplus \chi_y(z)$, each query can be answered (with zero error) by one bit of communication in either direction.

Put in the language of our reduction theorem, $\Psi = (P, S)$ with $P = \{u \in \{0,1\}^n : |u| = m\}^2$ and $S = \{\, (x,y) \in P : \ |x \cap y| \neq 0 \,\}$; while $\ell = 2^n$, $\mathcal{P} = \mathsf{PAR}^n_{2m} \subseteq 2^\ell$; and $F\colon \{0,1\}^{2n} \to \{0,1\}^\ell$ maps (x,y) to the truth table of $\chi_{x\oplus y}$. Since any two distinct parities are at distance $\frac{1}{2}$, we can take any $\varepsilon \le \frac{1}{2}$. We then have $B' = 1$, and by the theorem above we know that $\mathrm{CC}^{(k+2)}_{1/3}(\Psi) = \Omega\Big(m \log^{(k+2)} m\Big)$ for any $0 \le k \le \log^* m - 2$. Invoking Theorem 27 concludes the proof. ◀

We also outline below how Theorem 27 enables us to establish directly the lower bound part of Theorem 5, without relying on the transference theorem for LDTs.

Alternate proof of item (2) of Theorem 5. We start from the same communication complexity problem, "pointer-following," as in Section 4.3. Recall that an input to this problem is a pair of mappings (χ_A, χ_B) with $\chi_A\colon V_A \to V_B$, $\chi_B\colon V_B \to V_A$ (where V_A, V_B are two disjoint sets $\{v_0, \dots, v_{n/2-1}\}$ and $\{v_{n/2}, \dots, v_{n-1}\}$ of nodes of cardinality $n/2$). The function to compute is the indicator of the event where the vertex $v_i = \chi^{(k+2)}(v_0)$, reached by following the path of length $k+2$ starting at v_0, has an even index i.

We define $\Psi = (P, S)$ by setting $P \stackrel{\text{def}}{=} \{\, (\chi_A, \chi_B) : \ \chi_A\colon V_A \to V_B, \chi_B\colon V_B \to V_A \,\}$ and $S \stackrel{\text{def}}{=} \big\{\, (\chi_A, \chi_B) \in P : \ \chi^{(k+2)}(v_0) \text{ is an even-index node} \,\big\}$. The property is, as in Theorem 5, the subset of codewords $\mathcal{C}_{f_{k+1}}$ corresponding to the function $f_{k+1}\colon \mathbb{F}_n^n \to \{0,1\}$ of Section 4 (for a good code $C\colon \mathbb{F}_n^n \to \mathbb{F}_n^m$ as in Section 4.4.2).

Identifying $V \stackrel{\text{def}}{=} V_A \cup V_B = \{v_0, \dots, v_{n-1}\}$ with $\mathbb{F}_n$ in the natural way, we define the mapping $F\colon P \to \mathbb{F}_n^n$ by

$$F(\chi_A, \chi_B) = (\chi_A(v_0), \chi_A(v_1) \dots, \chi_A(n/2-1), \chi_B(n/2), \dots, \chi_B(n-1)).$$

From there, it is easy to check that by construction, (i) $(\chi_A, \chi_B) \in P \cap S$ implies $F(\chi_A, \chi_B) \in \mathcal{C}_{f_{k+1}}$, while (ii) $(\chi_A, \chi_B) \in P \setminus S$ implies that $F(\chi_A, \chi_B)$ is ε_0-far from $\mathcal{C}_{f_{k+1}}$, for some constant $\varepsilon_0 > 0$ depending on the code C. Observing that $B' \stackrel{\text{def}}{=} \max_{i \in [\ell]} \max(\overrightarrow{\mathrm{CC}}_0(F_i), \overleftarrow{\mathrm{CC}}_0(F_i)) =$

Open Problem 1, although one rather weak quantitatively. It also, as a special case, would separate adaptive and non-adaptive testing of m-linearity for $m = o(n)$, a longstanding open question [10, 6].

$O(\log n)$ (as each F_i is specified by $O(\log n)$ bits), we can invoke Theorem 27 along with the communication complexity lower bound of Theorem 9 to obtain that $Q^{(k)}_{1/3}(\varepsilon_0, \mathcal{C}_{f_{k+1}}) \geq \frac{1}{B'+1}\,\mathrm{CC}^{(k+2)}_{1/3}(\Psi) = \Omega\Big(\frac{n}{k^2 \log n}\Big)$. ◀

6.3 On the Relative Power of Round-Adaptive and Tail-Adaptive Testers

In this section, we show that the two notions of round- and tail-adaptive testers we introduced are not equivalent. As mentioned in Section 3, while round-adaptive testers are at least as powerful as tail-adaptive ones, there exist properties for which the separation is strict:

▶ **Theorem 29.** *Fix any $\alpha \in (0,1)$. There exists a constant $\beta \in (0,1)$ such that, for every $n \in \mathbb{N}$, the following holds. For every integer $0 \leq k \leq n^\beta$, there exists a property $\mathcal{P}_k \subseteq \mathbb{F}_n^{n^{1+\alpha}}$ such that, for any constant $\varepsilon \in (0,1]$,*

1. *there exists a $(k, \tilde{O}(k))$-round-adaptive (one-sided) tester for $\mathcal{P}_k$; yet*
2. *any (k,q)-tail-adaptive (two-sided) tester for $\mathcal{P}_k$ must satisfy $q = \Omega(n)$.*

Proof Sketch. The argument is very similar to that of Theorem 5, and follows the same overall structure. Namely, we slightly modify the k-iterated function f_k of Section 4 (which *was* computable by a $(k, k+1)$-tail-adaptive algorithm) to rule out tail-adaptive algorithms but not round-adaptive ones: that is, we define the function $f'_k \colon \mathbb{F}_n^n \to \mathbb{F}_n$ by

$$f'_k(x) = \begin{cases} 1 & \text{if } x_{x,g_{k-1}(x)} = x_{x,g_{k-1}(x)+1 \bmod n} \\ 0 & \text{otherwise.} \end{cases}$$

(Perhaps more clearly, f'_k is computed by iterating the pointer function k times, and then checking if the value x_i at the final coordinate $i \in [n]$ reached, and the value x_{i+1} at the adjacent coordinate $i+1$, are equal.) It is not hard to see that the counterparts of Claim 7 and Lemma 8 still hold for f'_k: first, the function is still easy to compute by $(k, k+2)$-round-adaptive algorithms. However, because the very last round requires 2 queries and not one (to query x_i and x_{i+1}, once the value of $i = g_{k-1}(x)$ has been obtained), tail-round-adaptive algorithms are no longer able to leverage this, and analogously to Lemma 8 we can conclude that there is no $(k, o(n/(k^2 \log n)))$-round-adaptive (randomized) LDT algorithm which computes f'_k. It then only remains to lift this DT separation to property testing: we can do this as before (noting, in the case of lifting the lower bound, that the reduction of Lemma 14 preserves the number of queries per round, and thus the "tailness" of the algorithm). ◀

Acknowledgments. We are grateful to Oded Goldreich for suggesting cycle freeness as a candidate natural property for proving an adaptivity hierarchy theorem, as well as for enlightening conversations that significantly contributed to this work; and wish to thank Rocco Servedio for helpful comments on an earlier version of this paper.

References

1 Scott Aaronson and Avi Wigderson. Algebrization: a new barrier in complexity theory. In *Proceedings of STOC*, pages 731–740, 2008. doi:10.1145/1374376.1374481.

2 Noga Alon, Eldar Fischer, Michael Krivelevich, and Mario Szegedy. Efficient testing of large graphs. *Combinatorica*, 20(4):451–476, 2000. doi:10.1007/s004930070001.

3 Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil P. Vadhan. Robust PCPs of Proximity, Shorter PCPs, and Applications to Coding. *SIAM Journal on Computing*, 36(4):889–974, 2006. doi:10.1137/S0097539705446810.

4 Eli Ben-Sasson, Prahladh Harsha, and Sofya Raskhodnikova. Some 3CNF properties are hard to test. *SIAM J. Comput.*, 35(1):1–21, 2005. doi:10.1137/S0097539704445445.

5 Arnab Bhattacharyya and Yuichi Yoshida. *Property Testing*. Forthcoming, 2017. URL: https://propertytestingbook.wordpress.com/.

6 Abhishek Bhrushundi, Sourav Chakraborty, and Raghav Kulkarni. Property testing bounds for linear and quadratic functions via parity decision trees. In *CSR*, volume 8476 of *Lecture Notes in Computer Science*, pages 97–110. Springer, 2014.

7 Eric Blais. Improved bounds for testing juntas. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*, pages 317–330. Springer, 2008.

8 Eric Blais. Testing juntas nearly optimally. In *Proceedings of STOC*, pages 151–158. ACM, 2009.

9 Eric Blais, Joshua Brody, and Kevin Matulef. Property testing lower bounds via communication complexity. *Computational Complexity*, 21(2):311–358, 2012. doi:10.1007/s00037-012-0040-x.

10 Eric Blais and Daniel M. Kane. Tight bounds for testing k-linearity. In *Proceedings of APPROX-RANDOM*, volume 7408 of *Lecture Notes in Computer Science*, pages 435–446. Springer, 2012.

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.

12 Joshua Brody, Kevin Matulef, and Chenggang Wu. Lower bounds for testing computability by small width OBDDs. In *TAMC*, volume 6648 of *Lecture Notes in Computer Science*, pages 320–331. Springer, 2011.

13 Harry Buhrman and Ronald de Wolf. Complexity measures and decision tree complexity: a survey. *Theor. Comput. Sci.*, 288(1):21–43, 2002. doi:10.1016/S0304-3975(01)00144-X.

14 Harry Buhrman, David García-Soriano, Arie Matsliah, and Ronald de Wolf. The non-adaptive query complexity of testing k-parities. *Chicago J. Theor. Comput. Sci.*, 2013, 2013.

15 Clément L. Canonne. A Survey on Distribution Testing: your data is Big. But is it Blue? *Electronic Colloquium on Computational Complexity (ECCC)*, 22:63, April 2015.

16 Xi Chen, Rocco A. Servedio, Li-Yang Tan, Erik Waingarten, and Jinyu Xie. Settling the query complexity of non-adaptive junta testing. In *Computational Complexity Conference (CCC)*, 2017. To appear.

17 Dingzhu Du and Frank K. Hwang. *Combinatorial Group Testing and Its Applications*. Applied Mathematics. World Scientific, 2000. URL: https://books.google.com/books?id=KW5-CyUUOggC, doi:10.1142/4252.

18 Oded Goldreich, editor. *Property Testing – Current Research and Surveys [outgrow of a workshop at the Institute for Computer Science (ITCS) at Tsinghua University, January 2010]*, volume 6390 of *Lecture Notes in Computer Science*. Springer, 2010. doi:10.1007/978-3-642-16367-8.

19 Oded Goldreich. On the communication complexity methodology for proving lower bounds on the query complexity of property testing. *Electronic Colloquium on Computational Complexity (ECCC)*, 20:73, 2013.

20 Oded Goldreich. *Introduction to Property Testing*. Forthcoming, 2017. URL: http://www.wisdom.weizmann.ac.il/~oded/pt-intro.html.

21 Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, 45(4):653–750, July 1998.

22 Oded Goldreich, Tom Gur, and Ilan Komargodski. Strong locally testable codes with relaxed local decoders. In *30th Conference on Computational Complexity (CCC 2015)*, volume 33 of *LIPIcs*, pages 1–41. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2015. doi:10.4230/LIPIcs.CCC.2015.1.

23 Oded Goldreich and Dana Ron. On testing expansion in bounded-degree graphs. *Electronic Colloquium on Computational Complexity (ECCC)*, 7:20, 2000.

24 Oded Goldreich and Dana Ron. Algorithmic aspects of property testing in the dense graphs model. *SIAM J. Comput.*, 40(2):376–445, 2011. doi:10.1137/090749621.

25 Oded Goldreich and Luca Trevisan. Three theorems regarding testing graph properties. *Random Struct. Algorithms*, 23(1):23–57, 2003. doi:10.1002/rsa.10078.

26 Tom Gur and Ron D. Rothblum. A hierarchy theorem for interactive proofs of proximity. 2017. The 8th Innovations in Theoretical Computer Science (ITCS 2017) conference (to appear).

27 Piotr Indyk, Eric Price, and David P. Woodruff. On the power of adaptivity in sparse recovery. In *Proceedings of FOCS*, pages 285–294, 2011. doi:10.1109/FOCS.2011.83.

28 Kevin Matulef, Ryan O'Donnell, Ronitt Rubinfeld, and Rocco A. Servedio. Testing ±1-weight halfspace. In *Proceedings of APPROX-RANDOM*, volume 5687 of *Lecture Notes in Computer Science*, pages 646–657. Springer, 2009.

29 Noam Nisan and Avi Wigderson. Rounds in communication complexity revisited. *SIAM Journal on Computing*, 22(1):211–219, February 1993. doi:10.1137/0222016.

30 Christos H. Papadimitriou and Michael Sipser. Communication complexity. In *Proceedings of STOC*, Proceedings of STOC, pages 196–200, New York, NY, USA, 1982. ACM. doi:10.1145/800070.802192.

31 Sofya Raskhodnikova and Adam D. Smith. A note on adaptivity in testing properties of bounded degree graphs. *Electronic Colloquium on Computational Complexity (ECCC)*, 13(089), 2006. URL: http://eccc.hpi-web.de/eccc-reports/2006/TR06-089/index.html.

32 Dana Ron. Property testing: A learning theory perspective. *Foundations and Trends in Machine Learning*, 1(3):307–402, 2008. doi:10.1561/2200000004.

33 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.

34 Dana Ron and Rocco A. Servedio. Exponentially improved algorithms and lower bounds for testing signed majorities. In *Proceedings of SODA*, pages 1319–1336. SIAM, 2013.

35 Dana Ron and Gilad Tsur. Testing computability by width-two OBDDs. *Theor. Comput. Sci.*, 420:64–79, 2012.

36 Ronitt Rubinfeld and Madhu Sudan. Robust characterization of polynomials with applications to program testing. *SIAM Journal on Computing*, 25(2):252–271, 1996.

37 Mert Sağlam and Gábor Tardos. On the communication complexity of sparse set disjointness and exists-equal problems. In *Proceedings of FOCS*, pages 678–687. IEEE Computer Society, 2013.

38 Rocco A. Servedio, Li-Yang Tan, and John Wright. Adaptivity helps for testing juntas. In *30th Conference on Computational Complexity (CCC 2015)*, volume 33 of *LIPIcs*, pages 264–279. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2015. doi:10.4230/LIPIcs.CCC.2015.264.

39 Roei Tell. Deconstructions of reductions from communication complexity to property testing using generalized parity decision trees. *Electronic Colloquium on Computational Complexity (ECCC)*, 21:115, 2014.

Distribution Testing Lower Bounds via Reductions from Communication Complexity

Eric Blais[1], Clément L. Canonne[2], and Tom Gur[3]

1 **University of Waterloo, Waterloo, ON, Canada**
eric.blais@uwaterloo.ca
2 **Columbia University, New York, NY, USA**
ccanonne@cs.columbia.edu
3 **UC Berkeley, Berkeley, CA, USA**
tom.gur@berkeley.edu

Abstract

We present a new methodology for proving distribution testing lower bounds, establishing a connection between distribution testing and the simultaneous message passing (SMP) communication model. Extending the framework of Blais, Brody, and Matulef [15], we show a simple way to reduce (private-coin) SMP problems to distribution testing problems. This method allows us to prove new distribution testing lower bounds, as well as to provide simple proofs of known lower bounds.

Our main result is concerned with testing identity to a specific distribution p, given as a parameter. In a recent and influential work, Valiant and Valiant [53] showed that the sample complexity of the aforementioned problem is closely related to the $\ell_{2/3}$-quasinorm of p. We obtain alternative bounds on the complexity of this problem in terms of an arguably more intuitive measure and using simpler proofs. More specifically, we prove that the sample complexity is essentially determined by a fundamental operator in the theory of interpolation of Banach spaces, known as Peetre's *K-functional*. We show that this quantity is closely related to the size of the effective support of p (loosely speaking, the number of supported elements that constitute the vast majority of the mass of p). This result, in turn, stems from an unexpected connection to functional analysis and refined concentration of measure inequalities, which arise naturally in our reduction.

1998 ACM Subject Classification F.1.3 [Computation by Abstract Devices] Complexity Measures and Classes

Keywords and phrases Distribution testing, communication complexity, lower bounds, simultaneous message passing, functional analysis

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.28

1 Introduction

Distribution testing, as first explicitly introduced in [9], is a branch of property testing [48, 31] concerned with the study of sublinear algorithms for making approximate decisions regarding probability distributions over massive domains. These algorithms are granted access to independent samples from an unknown distribution and are required to *test* whether this distribution has a certain global property. That is, a tester for property Π of distributions over domain Ω receives a proximity parameter $\varepsilon > 0$ and is asked to determine whether a distribution p over Ω (denoted $p \in \Delta(\Omega)$) has the property Π or is ε-far (say, in ℓ_1-distance) from any distribution that has Π, using a small number of independent samples from p. The *sample complexity* of Π is then the minimal number of samples needed to test it. Throughout

32nd Computational Complexity Conference (CCC 2017).
Editor: Ryan O'Donnell; Article No. 28; pp. 28:1–28:40


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

the introduction, we fix ε to be small constant and refer to a tester with respect to proximity parameter ε as an *ε-tester*.

In recent years, distribution testing has been studied extensively. In a significant body of work spanning more than a decade [8, 41, 7, 42, 2, 14, 37, 53, 23, 56, 36, 13, 25], a myriad of properties has been investigated under this lens. Starting with [32, 10, 8], this includes the testing of symmetric properties [45, 54, 51, 52], of structured families [11, 35, 3, 17, 4, 20, 19], as well as testing under some assumption on the unknown instance [47, 24, 27, 26]. Tight upper and lower bounds on the sample complexity have been obtained for properties such as uniformity, identity to a specified distribution, monotonicity, and many more (see references above, or [46, 18] for surveys). However, while by now numerous techniques and approaches are available to design distribution testers, our arsenal of tools for proving lower bounds on the sample complexity of distribution testing is significantly more limited. There are only a handful of standard techniques to prove lower bounds; and indeed the vast majority of the lower bounds in the literature are shown via *Le Cam's two-point method* (also known as the "easy direction" of Yao's minimax principle) [57, 44]. In this method, one first defines two distributions $\mathcal{Y}$ and $\mathcal{N}$ over distributions that are respectively yes-instances (having the property) and no-instances (far from having the property). Then it remains to show that with high probability over the choice of the instance, every tester that can distinguish between $p^{\mathsf{yes}} \sim \mathcal{Y}$ and $p^{\mathsf{no}} \sim \mathcal{N}$ must use at least a certain number of samples. In view of this scarcity, there has been in recent years a trend towards trying to obtain more, or simpler to use, techniques [54, 25]; however, this state of affairs largely remains the same.

In this work, we reveal a connection between distribution testing and the simultaneous message passing (SMP) communication model, which in turn leads to a new methodology for proving distribution testing lower bounds. Recall that in a private-coin SMP protocol, Alice and Bob are given strings $x, y \in \{0,1\}^k$ (respectively), and each of the players is allowed to send a message to a referee (which depends on the player's input and private randomness) who is then required to decide whether $f(x,y) = 1$ by only looking at the players' messages and flipping coins.

Extending the framework of Blais, Brody, and Matulef [15], we show a simple way of reducing (private-coin) SMP problems to distribution testing problems. This foregoing methodology allows us to prove new distribution testing lower bounds, as well as to provide simpler proofs of known lower bounds for problems such as testing uniformity, monotonicity, and k-modality (see Section 8).

Our main result is a characterization of the sample complexity of the distribution identity testing problem in terms of a key operator in the study of interpolation spaces, which arises naturally from our reduction and for which we are able to provide an intuitive interpretation. Recall that in this problem, the goal is to determine whether a distribution q over domain Ω (denoted $q \in \Delta(\Omega)$) is identical to a fixed distribution p; that is, given a full description of $p \in \Delta(\Omega)$, we ask how many independent samples from q are needed to decide whether $q = p$, or whether q is ε-far in ℓ_1-distance from p.[1]

In a recent and influential work, Valiant and Valiant [53] showed that the sample complexity of the foregoing question is closely related to the $\ell_{2/3}$-quasinorm of p, defined as $\|p\|_{2/3} = \left(\sum_{\omega\in\Omega} |p(\omega)|^{2/3}\right)^{3/2}$. That is, viewing a distribution $p \in \Delta(\Omega)$ as an $|\Omega|$-dimensional vector of probabilities, let $p_{-\varepsilon}^{-\max}$ be the vector obtained from p by zeroing

[1] Note that this is in fact a family of massively parameterized properties $\{\Pi_p\}_{p\in\Delta(\Omega)}$, where Π_p is the property of being identical to p. See [39] for an excellent survey concerning massively parameterized properties.

Table 1 Summary of results. All the bounds are stated for constant proximity parameter ε.

Property	Our results	Previous bounds
Uniformity	$\tilde{\Omega}(\sqrt{n})$	$\Theta(\sqrt{n})$ [32, 42]
Identity to p	$\Omega\big(\kappa_p^{-1}(1-\varepsilon)\big), O\big(\kappa_p^{-1}(1-c\cdot\varepsilon)\big)$	$\Omega\Big(\|p_{\varepsilon}^{-\max}\|_{2/3}\Big), O\Big(\|p_{c'\cdot\varepsilon}^{-\max}\|_{2/3}\Big)$ [53]
Monotonicity	$\tilde{\Omega}(\sqrt{n})$	$\Theta(\sqrt{n})$ [11, 4, 20]
k-modal	$\tilde{\Omega}(\sqrt{n})$	$\tilde{\Omega}(\max(\sqrt{n}, k))$ [19]
Log-concavity, Monotone Hazard Rate	$\tilde{\Omega}(\sqrt{n})$	$\Theta(\sqrt{n})$ [4, 20]
Binomial, Poisson Binomial	$\tilde{\Omega}\big(n^{1/4}\big)$	$\Theta\big(n^{1/4}\big)$ ([3, 20]
Symmetric sparse support	$\tilde{\Omega}(\sqrt{n})$	
Junta distributions (PAIRCOND model)	$\Omega(k)$	

out the largest entry as well as the set of smallest entries summing to ε (note that $p_{-\varepsilon}^{-\max}$ is no longer a probability distribution). Valiant and Valiant gave an ε-tester for testing identity to p with sample complexity $O\big(\|p_{-c\varepsilon}^{-\max}\|_{2/3}\big)$, where $c > 0$ is a universal constant, and complemented this result with a lower bound of $\Omega\big(\|p_{-\varepsilon}^{-\max}\|_{2/3}\big)$.[2]

In this work, using our new methodology, we show alternative and similarly tight bounds on the complexity of identity testing, in terms of a more intuitive measure (as we discuss below) and using simpler arguments. Specifically, we prove that the sample complexity is essentially determined by a fundamental quantity in the theory of interpolation of Banach spaces, known as Peetre's *K-functional*. Formally, for a distribution $p \in \Delta(\Omega)$, the K-functional between ℓ_1 and ℓ_2 spaces is the operator defined for $t > 0$ by

$$\kappa_p(t) = \inf_{p'+p''=p} \|p'\|_1 + t\|p''\|_2.$$

This operator can be thought of as an interpolation norm between the ℓ_1 and ℓ_2 norms of the distribution p (controlled by the parameter t), naturally inducing a partition of p into two sub-distributions: p', which consists of "heavy hitters" in ℓ_1-norm, and p'', which has a bounded ℓ_2-norm. Indeed, the approach of isolating elements with large mass and testing in ℓ_2-norm seems inherent to the problem of identity testing, and is the core component of both early works [32, 8] and more recent ones [27, 25, 30]. As a further connection to the identity testing question, we provide an easily interpretable proxy for this measure κ_p, showing that the K-functional between the ℓ_1 and ℓ_2 norms of the distribution p is closely related to the size of the effective support of p, which is the number of supported elements that constitute the vast majority of the mass of p; that is, we say that p has ε-*effective support* of size T if $1 - O(\varepsilon)$ of the mass of p is concentrated on T elements (see Section 2.4 for details).

Having defined the K-functional, we can proceed to state the lower bound we derive for the problem.[3]

[2] We remark that for certain p's, the asymptotic behavior of $O\big(\|p_{-c\varepsilon}^{-\max}\|_{2/3}\big)$ strongly depends on the constant c, and so it cannot be omitted from the expression. We further remark that this result was referred to by Valiant and Valiant as "instance-optimal identity testing" as the resulting bounds are phrased as a function of the distribution p itself – instead of the standard parameter which is the domain size n.

[3] As stated, this result is a slight strengthening of our communication complexity reduction, which yields

▶ **Theorem 1** (Informally stated). *Any ε-tester of identity to $p \in \Delta(\Omega)$ must have sample complexity $\Omega\big(\kappa_p^{-1}(1-2\varepsilon)\big)$.*

In particular, straightforward calculations show that for the uniform distribution we obtain a tight lower bound of $\Omega(\sqrt{n})$, and for the Binomial distribution we obtain a tight lower bound of $\Omega\big(n^{1/4}\big)$.

To show that tightness of the lower bound above, we complement it with a nearly matching upper bound, also expressed in terms of the K-functional.

▶ **Theorem 2** (Informally stated). *There exist an absolute constant $c > 0$ and an ε-tester of identity to $p \in \Delta(\Omega)$ that uses $O\big(\kappa_p^{-1}(1-c\varepsilon)\big)$ mples.*[4]

In the following section, we provide an overview of our new methodology as well as the proofs for the above theorems. We also further discuss the interpretability of the K-functional and show its close connection to the effective support size. We conclude this section by outlining a couple of extensions of our methodology.

Dealing with sub-constant values of the proximity parameter. Similarly to the communication complexity methodology for proving property testing lower bounds [15], our method inherently excels in the regime of *constant* values of the proximity parameter ε. Therefore, in this work we indeed focus on the constant proximity regime. However, in Section 5.1 we demonstrate how to obtain lower bounds that asymptotically increase as ε tends to zero, via an extension of our general reduction.

Extending the methodology to testing with conditional samples. Testers with sample access are by far the most commonly studied algorithms for distribution testing. However, many scenarios that arise both in theory and practice are not fully captured by this model. In a recent line of works [22, 21, 1, 28, 29], testers with access to *conditional* samples were considered, addressing situations in which one can control the samples that are obtained by requesting samples conditioned on membership on subsets of the domain. In Section 9, we give an example showing that it is possible to extend our methodology to obtain lower bounds in the conditional sampling model.

1.1 Organization

We first give a technical overview in Section 2, demonstrating the new methodology and presenting our bounds on identity testing. Section 3 then provides the required preliminaries for the main technical sections. In Section 4 we formally state and analyze the SMP reduction methodology for proving distribution testing lower bounds. In Section 5, we instantiate the basic reduction, obtaining a lower bound on uniformity testing, and in Section 5.1 show how to extend the methodology to deal with sub-constant values of the proximity parameter. (We stress that Section 5.1 is *not* a prerequisite for the rest of the sections, and can be skipped at the reader's convenience.) In Section 6 we provide an exposition to the K-functional and generalize inequalities that we shall need for the following sections. Section 7 then contains the proofs of both lower and upper bounds on the problem of identity testing, in terms of

a lower bound of $\Omega\big(\kappa_p^{-1}(1-2\varepsilon)/\log n\big)$. This strengthening is described in Section 7.3.

[4] Similarly to the [53] bound, for certain p's, the asymptotic behavior of $O\big(\kappa_p^{-1}(1-2\varepsilon)\big)$ depends on the constant c, and so it cannot be omitted from the expressi

the K-functional. In Section 8, we demonstrate how to easily obtain lower bounds for other distribution testing problems. Finally, in Section 9 we discuss extensions to our methodology; specifically, we explain how to obtain lower bounds in various metrics, and show a reduction from communication complexity to distribution testing in the conditional sampling model.

2 Technical Overview

In this section we provide an overview of the proof of our main result, which consists of new lower and upper bounds on the sample complexity of testing identity to a given distribution, expressed in terms of an intuitive, easily interpretable measure. To do so, we first introduce the key component of this proof, the methodology for proving lower bounds on distribution testing problems via reductions from SMP communication complexity. We then explain how the relation to the theory of interpolation spaces and the so-called K-functional naturally arises when applying this methodology to the identity testing problem.

For the sake of simplicity, throughout the overview we fix the domain $\Omega = [n]$ and fix the proximity parameter ε to be a small constant. We begin in Section 2.1 by describing a simple "vanilla" reduction for showing an $\tilde{\Omega}(\sqrt{n})$ lower bound on the complexity of testing that a distribution is uniform. Then, in Section 2.2 we extend the foregoing approach to obtain a new lower bound on the problem of testing identity to a fixed distribution. This lower bound depends on the best rate obtainable by a special type of error-correcting codes, which we call *p-weighted codes*. In Section 2.3, we show how to relate the construction of such codes to concentration of measure inequalities for weighted sums of Rademacher random variables; furthermore, we discuss how the use of the *K-functional*, an interpolation norm between ℓ_1 and ℓ_2 spaces, leads to stronger concentration inequalities than the ones derived by Chernoff bounds or the central limit theorem. Finally, in Section 2.4 we establish nearly matching upper bounds for testing distribution identity in terms of this K-functional, using a proxy known as the Q-norm. We then infer that the sample complexity of testing identity to a distribution p is roughly determined by the size of the *effective support* of p (which is, loosely speaking, the number of supported elements which together account for the vast majority of the mass of p).

2.1 Warmup: Uniformity Testing

Consider the problem of testing whether a distribution $q \in \Delta([n])$ is the *uniform distribution*; that is, how many (independent) samples from q are needed to decide whether q is the uniform distribution over $[n]$, or whether q is ε-far in ℓ_1-distance from it. We reduce the SMP communication complexity problem of *equality* to the distribution testing problem of uniformity testing.

Recall that in a private-coin SMP protocol for equality, Alice and Bob are given strings $x, y \in \{0,1\}^k$ (respectively), and each of the players is allowed to send a message to a referee (which depends on the player's input and private randomness) who is then required to decide whether $x = y$ by only looking at the players' messages and flipping coins.

The reduction is as follows. Assume there exists a uniformity tester with sample complexity s. Each of the players encodes its input string via a balanced asymptotically good code C (that is, $C\colon \{0,1\}^k \to \{0,1\}^n$ is an error-correcting code with constant rate and relative distance $\delta = \Omega(1)$, which satisfies the property that each codeword of C contains the same number of 0's and 1's). Denote by $A \subset [n]$ the locations in which $C(x)$ takes the value 1 (i.e., $A = \{\, i \in [n] : C(x)_i = 1 \,\}$), and denote by $B \subset [n]$ the locations in which $C(y)$ takes the value 0 (i.e., $B = \{\, i \in [n] : C(y)_i = 0 \,\}$). Alice and Bob each send $O(s)$ uniformly

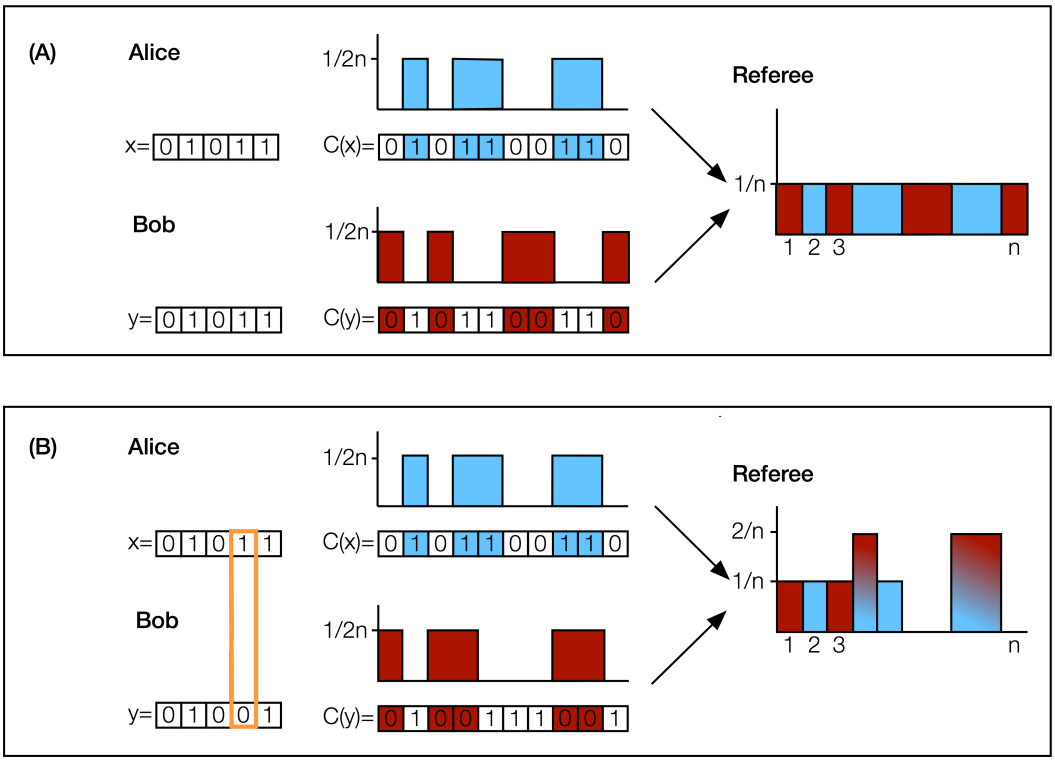


Figure 1 The reduction from equality in the SMP model to uniformity testing of distributions. In (A) we see that the uniform distribution is obtained when $x = y$, whereas in (B) we see that when $x \neq y$, we obtain a distribution that is "far" from uniform.

distributed samples from A and B, respectively. Finally, the referee invokes the uniformity tester with respect to the distribution $q = (\mathcal{U}_A + \mathcal{U}_B)/2$, emulating each draw from q by tossing a random coin and deciding accordingly whether to use a sample by Alice or Bob. See Figure 1.

The idea is that if $x = y$, then $C(x) = C(y)$, and so A and B are a *partition* of the set $[n]$. Furthermore, since $|C(x)| = |C(y)| = n/2$, this is a equipartition. Now, since Alice and Bob send uniform samples from an equipartition of $[n]$, the distribution q that the referee emulates is in fact the uniform distribution over $[n]$, and so the uniformity tester will accept. On the other hand, if $x \neq y$, then $C(x)$ and $C(y)$ disagree on a constant fraction of the domain. Thus, A and B intersect on $\delta/2$ elements, as well as do not cover $\delta/2$. Therefore q is uniform on a $(1-\delta)$-fraction of the domain, unsupported on a $(\delta/2)$-fraction of the domain, and has "double" weight $2/n$ on the remaining $(\delta/2)$-fraction. In particular, since $\delta = \Omega(1)$, the emulated distribution q is $\Omega(1)$-far (in ℓ_1-distance) from uniform, and it will be rejected by the uniformity tester.

As each sample sent by either Alice or Bob was encoded with $O(\log n)$ bits, the above constitutes an SMP protocol for equality with communication complexity $O(s \log(n))$. Yet it is well known [40] that the players must communicate $\Omega(\sqrt{k})$ bits to solve this problem (see Section 4), and so we deduce that $s = \Omega(\sqrt{k}/\log(n)) = \tilde{\Omega}(\sqrt{n})$.

2.2 Revisiting Distribution Identity Testing: A New Lower Bound

Next, consider the problem of testing whether a distribution $q \in \Delta([n])$ is identical to a fixed distribution p, provided as a (massive) parameter; that is, given a full description of $p \in \Delta([n])$, we ask how many independent samples from q are needed to decide whether $q = p$, or whether q is ε-far in ℓ_1-distance from p. As mentioned earlier, Valiant and Valiant [53] established both upper and lower bounds on this problem, involving the $\ell_{2/3}$-quasinorm of p. We revisit this question, and show different – and more interpretable – upper and lower bounds. First, by applying our new communication complexity methodology to the distribution identity problem, we obtain a simple lower bound expressed in terms of a new parameter, which is closely related to the *effective support size* of p.

Consider any fixed $p \in \Delta([n])$. As a first idea, it is tempting to reduce equality in the SMP model to testing identity to p by following the uniformity reduction described in Section 2.1, only instead of having Alice and Bob send *uniform* samples from A and B, respectively, we have them send samples from p *conditioned* on membership in A and B respectively. That is, as before Alice and Bob encode their inputs x and y via a balanced, asymptotically good code C to obtain the sets $A = \{\, i \in [n] : C(x)_i = 1 \,\}$ and $B = \{\, i \in [n] : C(y)_i = 0 \,\}$, which partition $[n]$ if $x = y$, and intersect on $\Omega(n)$ elements (as well as fail to cover $\Omega(n)$ elements of $[n]$) if $x \neq y$. Only now, Alice sends samples independently drawn from $p|_A$, i.e., p conditioned on the samples belonging to A, and Bob sends samples independently drawn from $p|_B$, i.e., p conditioned on the samples belonging to B; and the referee emulates the distribution $q = (p|_A + p|_B)/2$.

However, two problems arise in the foregoing approach. The first is that while indeed when $x = y$ the reduction induces an equipartition A, B of the domain, the resulting weights $p(A)$ and $p(B)$ in the mixture may still be dramatically different, in which case the referee will need much more samples from one of the parties to emulate p. The second is a bit more subtle, and has to do with the fact that the properties of this partitioning are with respect to the *size* of the symmetric difference $A \Delta B$, while really we are concerned about its *mass* under the emulated distribution q (and although both are proportional to each other in the case of the uniform distribution, for general p we have no such guarantee). Namely, when $x \neq y$ the domain elements which are responsible for the distance from p (that is, the elements which are covered by both parties ($A \cap B$) and by neither of the parties ($[n] \setminus (A \cup B)$) may only have a small mass according to p, and thus the emulated distribution q will not be sufficiently far from p. A natural attempt to address these two problems would be to preprocess p by discarding its light elements, focusing only on the part of the domain where p puts enough mass pointwise; yet this approach can also be shown to fail, as in this case the reduction may still not generate enough distance.[5]

Instead, we take a different route. The key idea is to consider a new type of codes, which we call *p-weighted codes*, which will allow us to circumvent the second obstacle. These are code whose distance guarantee is weighted according to the distribution p; that is, instead of requiring that every two codewords c, c' in a code C satisfy $\mathrm{dist}(x, y) \stackrel{\text{def}}{=} \sum_{i=1}^{n} |x_i - y_i| \geq \delta$,

[5] In more detail, this approach would consider the distribution p' obtained by iteratively removing the lightest elements of p until a total of ε probability mass was removed. This way, every element i in the support of p' is guaranteed to have mass $p'_i \geq \varepsilon/n$: this implies that the weights $p'(A)$ and $p'(B)$ are proportional, and that each element that is either covered by both parties or not covered at all will contribute ε/n to the distance from p'. However, the total distance of q from p would only be $\Omega\big(|\operatorname{supp}(p')| \cdot \varepsilon/n\big)$; and this only suffices if p and p' have comparable support size, i.e. $|\operatorname{supp}(p)| = O\big(|\operatorname{supp}(p')|\big)$.

we consider a code $C_p \colon \{0,1\}^k \to \{0,1\}^n$ such that every $c, c' \in C_p$ satisfy

$$\mathrm{dist}_p(x,y) \stackrel{\text{def}}{=} \sum_{i=1}^{n} p(i) \cdot |x_i - y_i| \geq \delta.$$

Furthermore, to handle the first issue, we adapt the "balance" property accordingly, requiring that each codeword be balanced according to p, that is, every $c \in C_p$ satisfies $\sum_{i=1}^{n} p(i) \cdot c_i = 1/2$.

It is straightforward to see that if we invoke the above reduction while letting the parties encode their inputs via a balance p-weighted code C_p, then both of the aforementioned problems are resolved; that is, by the p-balance property the weights $p(A)$ and $p(B)$ are equal, and by the p-distance of C_p we obtain that for $x \neq y$ the distribution $q = (p_A + p_B)/2$ is $\Omega(1)$-far from p. Hence we obtain a lower bound of $\Omega\Big(\sqrt{k}/\log(n)\Big)$ on the query complexity of testing identity to p. To complete the argument, it remains to construct such codes, and determine what the best rate k/n that can be obtained by p-weighted codes is.

2.3 Detour: p-weighted Codes, Peetre's K-functional, and beating the CLT

The discussion of previous section left us with the task of constructing high-rate p-weighted codes. Note that unlike standard (uniformly weighted) codes, for which we can easily obtain constant rate, there exist some p's for which high rate is impossible (for example, if $p \in \Delta([n])$ is only supported on one element, we can only obtain rate $1/n$). In particular, by the sphere packing bound, every p-weighted code $C \colon \{0,1\}^k \to \{0,1\}^n$ with distance δ must satisfy

$$\underbrace{2^k}_{\#\text{codewords}} \leq \frac{2^n}{\mathrm{Vol}_{\mathbb{F}_2^n,\mathrm{dist}_p}(\delta/2)},$$

where $\mathrm{Vol}_{\mathbb{F}_2^n,\mathrm{dist}_p}(r)$ is the volume of the p-ball of radius r in the n-dimensional hypercube, given by

$$\mathrm{Vol}_{\mathbb{F}_2^n,\mathrm{dist}_p}(r) \stackrel{\text{def}}{=} \left| \left\{ w \in \mathbb{F}_2^n \,:\, \sum_{i=1}^{n} p_i \cdot w_i \leq r \right\} \right|.$$

Hence, we must have $k \leq n - \log \mathrm{Vol}_{\mathbb{F}_2^n,\mathrm{dist}_p}(\delta/2)$.

In Section 7.1 we show that there exist (roughly) balanced p-weighted codes with nearly-optimal rate,[6] and so it remains to determine the volume of the p-ball of radius ε in the n-dimensional hypercube, where recall that ε is the proximity parameter of the test. To this end, it will be convenient to represent this quantity as a concentration inequality of sums of weighted Rademacher random variables, as follows

$$\mathrm{Vol}_{\mathbb{F}_2^n,\mathrm{dist}_p}(\varepsilon) = 2^n \Pr_{Y \sim \{0,1\}^n}\left[\sum_{i=1}^{n} p_i Y_i \leq \varepsilon\right] = 2^n \Pr_{X \sim \{-1,1\}^n}\left[\sum_{i=1}^{n} p_i X_i \geq 1 - 2\varepsilon\right]. \tag{1}$$

Applying standard tail bounds derived from the central limit theorem (CLT), we have that

$$\Pr_{X \sim \{-1,1\}^n}\left[\sum_{i=1}^{n} p_i X_i \geq 1 - 2\varepsilon\right] \leq e^{\frac{-(1-2\varepsilon)^2}{2\|p\|_2^2}}, \tag{2}$$

[6] We remark that since these codes are not perfectly p-balanced, a minor modification to the reduction needs to be done. See Section 7.1 for details.

and so we can obtain a p-weighted code $C_p\colon \{0,1\}^k \to \{0,1\}^n$ with dimension $k = O(1/\|p\|_2^2)$, which in turn, by the reduction described in Section 2.2, implies a lower bound of $\Omega(1/(\|p\|_2 \cdot \log(n)))$ on the complexity of testing identity to p.

Unfortunately, the above lower bound is not as strong as hoped, and in particular, far weaker than the $\|p^{-\max}_{-\varepsilon}\|_{2/3}$ bound of [53].[7] Indeed, it turns out that the CLT-based bound in Equation 2 is only tight for distributions satisfying $\|p\|_\infty = O(\|p\|_2^2)$, and is in general too crude for our purposes. Instead, we look for stronger concentration of measure inequalities that "beat" the CLT. To this end, we shall use powerful tools from the theory of interpolation spaces. Specifically, we consider Peetre's *K-functional* between ℓ_1 and ℓ_2 spaces. Loosely speaking, this is the operator defined for $t > 0$ by

$$\kappa_p(t) = \inf_{p'+p''=p} \|p'\|_1 + t\|p''\|_2.$$[8]

This K-functional can be thought of as an interpolation norm between the ℓ_1 and ℓ_2 norms of the distribution p (and accordingly, for any fixed t it defines a norm on the space $\ell_1 + \ell_2$). In particular, note that for large values of t the function $\kappa_p(t)$ is close to $\|p\|_1$, whereas for small values of t it will behave like $t\|p\|_2$.

The foregoing connection is due to Montgomery-Smith [38], who established the following concentration of measure inequality for weighted sums of Rademacher random variables,

$$\Pr\left[\sum_{i=1}^n p_i X_i \geq \kappa_p(t)\right] \leq e^{-\frac{t^2}{2}}. \tag{3}$$

Furthermore, he proved that this concentration bound is essentially tight (see Section 6 for a precise statement). Plugging (3) into (1), we obtain a lower bound of $\Omega(\kappa_p^{-1}(1-2\varepsilon)/\log(n))$ on the complexity of testing identity to p.

To understand and complement this result, we describe in the next subsection a nearly tight upper bound for this problem, also expressed in terms of this K-functional; implying that this unexpected connection is in fact not a coincidence, but instead capturing an intrinsic aspect of the identity testing question. We also give a natural interpretation of this bound, showing that the size of the *effective support* of p (roughly, the number of supported elements that constitute the vast majority of the mass of p) is a good proxy for this parameter $\kappa_p^{-1}(1-2\varepsilon)$ – and thus for the complexity of testing identity to p.

2.4 Using the Q-norm Proxy to Obtain an Upper Bound

To the end of obtaining an upper bound on the sample complexity of testing identity to p, in terms of the K-functional, it will actually be convenient to look at a related quantity, known as the *Q-norm* [38]. At a high-level, the Q-norm of a distribution p, for a given parameter $T \in \mathbb{N}$, is the maximum one can reach by partitioning the domain of p into T sets and taking

[7] For example, fix $\alpha \in (0,1)$, and consider the distribution $p \in \Delta([n])$ in which $n/2$ elements are of mass $1/n$, and $n^\alpha/2$ elements are of mass $1/n^\alpha$. It is straightforward to verify that $\|p\|_2^{-1} = \Theta\big((\sqrt{n})^\alpha\big)$, whereas $\|p\|_{2/3} = \Theta\big(\sqrt{n}\big)$. (Intuitively, this is because the ℓ_2-norm is mostly determined by the few heavy elements, whereas the $\ell_{2/3}$-quasinorm is mostly determined by the numerous light elements.)

[8] Interestingly, Holmstedt [34] showed that the infimum is *approximately* obtained by partitioning $p = (p', p'')$ such that p' consists of heaviest t^2 coordinates of p and p'' consists of the rest (for more detail, see Theorem 15).

the sum of the ℓ_2 norms of these T subvectors. That is

$$\|p\|_{Q(T)} \stackrel{\text{def}}{=} \sup \left\{ \sum_{j=1}^{T} \left(\sum_{i \in A_j} p_i^2 \right)^{1/2} \;:\; (A_j)_{1 \le j \le T} \text{ partition of } \mathbb{N} \right\}.$$

Astashkin [6], following up Montgomery-Smith [38], showed that the Q-norm constitutes a good approximation of K-functional, by proving that

$$\|p\|_{Q(t^2)} \le \kappa_p(t) \le \sqrt{2}\|p\|_{Q(t^2)}.$$

In Section 6 we further generalize this claim and show it is possible to get a tradeoff in the upper bound; specifically, we prove that $\kappa_p(t) \le \|p\|_{Q(2t^2)}$. Thus, it suffices to prove an upper bound on distribution identity testing in terms of the Q-norm.

From an algorithmic point of view, it is not immediately clear that switching to this Q-norm is of any help. However, we will argue that this value captures – in a very quantitative sense – the notion of the *sparsity* of p. As a first step, observe that if $\|p\|_{Q(T)} = 1$, then the distribution p is supported on at most T elements. To see this, denote by p_{A_j} the restriction of the sequence p to the indices in A_j, and note that if $\|p\|_{Q(T)} \stackrel{\text{def}}{=} \sum_{j=1}^{T} \|p_{A_j}\|_2 = 1$, then by the monotonicity of ℓ_p norms and since $\sum_{j=1}^{T} \|p_{A_j}\|_1 = \|p\|_1 = 1$ we have that

$$\sum_{j=1}^{T} (\underbrace{\|p_{A_j}\|_1 - \|p_{A_j}\|_2}_{\ge 0}) = 0,$$

which implies that $\|p_{A_j}\|_1 = \|p_{A_j}\|_2$ for all $j \in [T]$.

Now, it turns out that it is possible to obtain a *robust* version of the foregoing observation, yielding a sparsity lemma that, roughly speaking, shows thats if $\|p\|_{Q(T)} \ge 1 - \varepsilon$, then $1 - O(\varepsilon)$ of the mass of p is concentrated on T elements: in this case we say that p has $O(\varepsilon)$-*effective support* of size T. (See Lemma 31 for precise statement of the sparsity lemma.)

This property of the Q-norm suggests the following natural test for identity to a distribution p: Simply fix T such that $\|p\|_{Q(T)} = 1 - \varepsilon$, and apply one of the standard procedures for testing identity to a distribution with support size T, which require $O(\sqrt{T})$ samples. But by the previous discussion, we have $\|p\|_{Q(2t^2)} \ge \kappa_p(t)$, so that setting $T = 2t^2$ for the "right" choice of $t = \kappa_p^{-1}(1 - 2\varepsilon)$ will translate to an $O(t)$ upper bound – which is what we were aiming for.

3 Preliminaries

Notation. We write $[n]$ for the (ordered) set of integers $\{1, \ldots, n\}$, and ln, log for respectively the natural and binary logarithms. We use the notation $\tilde{\Omega}(f)$ to hide polylogarithmic dependencies on the argument, i.e. for expressions of the form $\Omega(f \log^c f)$ (for some absolute constant c). All throughout the paper, we denote by $\Delta(\Omega)$ the set of discrete probability distributions over domain Ω. When the domain is a subset of the natural numbers $\mathbb{N}$, we shall identify a distribution $p \in \Delta(\Omega)$ with the sequence $(p_i)_{i \in \mathbb{N}} \in \ell_1$ corresponding to its probability mass function (pmf). For a subset $S \subseteq \Omega$, we denote by $p|_S$ the normalized projection of p to S (so $p|_S$ is a probability distribution).

For an alphabet Σ, we denote the projection of $x \in \Sigma^n$ to a subset of coordinates $I \subseteq [n]$ by $x|_I$. For $i \in [n]$, we write $x_i = x|_{\{i\}}$ to denote the projection to a singleton. We denote the *relative Hamming distance*, over alphabet Σ, between two strings $x \in \Sigma^n$ and $y \in \Sigma^n$

by $\mathrm{dist}(x,y) \stackrel{\text{def}}{=} |\{\, x_i \neq y_i \,:\, i \in [n] \,\}|\,/n$. If $\mathrm{dist}(x,y) \leq \varepsilon$, we say that x is *ε-close* to y, and otherwise we say that x is *ε-far* from y. Similarly, we denote the *relative Hamming distance* of x from a non-empty set $S \subseteq \Sigma^n$ by $\mathrm{dist}(x,S) \stackrel{\text{def}}{=} \min_{y \in S} \mathrm{dist}(x,y))$. If $\mathrm{dist}(x,S) \leq \varepsilon$, we say that x is *ε-close* to S, and otherwise we say that x is *ε-far* from S.

Distribution Testing. A *property* of distributions over Ω is a subset $\mathcal{P} \subseteq \Delta(\Omega)$, consisting of all distributions that have the property. Given two distributions $p, q \in \Delta(\Omega)$, the ℓ_1 distance between p and q is defined as the ℓ_1 distance between their pmf's, namely $\|p-q\|_1 = \sum_{i \in \Omega} |p_i - q_i|$.[9] Given a property $\mathcal{P} \subseteq \Delta(\Omega)$ and a distribution $p \in \Delta(\Omega)$, we then define the distance of p to $\mathcal{P}$ as $\ell_1(p,\mathcal{P}) = \inf_{q \in \mathcal{P}} \|p-q\|_1$.

A *testing algorithm* for a fixed property $\mathcal{P}$ is then a randomized algorithm $\mathcal{T}$ which takes as input n, $\varepsilon \in (0,1]$, and is granted access to independent samples from an unknown distribution p; and satisfies the following.

1. if $p \in \mathcal{P}$, the algorithm outputs accept with probability at least $2/3$;
2. if $\ell_1(p,\mathcal{P}) \geq \varepsilon$, it outputs reject with probability at least $2/3$.

In other words, $\mathcal{T}$ must accept with high probability if the unknown distribution has the property, and reject if it is *ε-far* from having it. The *sample complexity* of the algorithm is the number of samples it draws from the distribution in the worst case.

Inequalities. We now state a standard probabilistic result that some of our proofs will rely on, the Paley–Zygmund anticoncentration inequality:

▶ **Theorem 3** (Paley–Zygmund inequality). *Let X be a non-negative random variable with finite variance. Then, for any $\theta \in [0,1]$,*

$$\Pr[\, X > \theta \mathbb{E}[X] \,] \geq (1-\theta)^2 \frac{\mathbb{E}[X]^2}{\mathbb{E}[X^2]}.$$

We will also require the following version of the rearrangement inequality, due to Hardy and Littlewood (cf. for instance [12, Theorem 2.2]):

▶ **Theorem 4** (Hardy–Littlewood Inequality). *Fix any $f, g \colon \mathbb{R} \to [0,\infty)$ such that $\lim_{\pm\infty} f = \lim_{\pm\infty} g = 0$. Then,*

$$\int_{\mathbb{R}} fg \leq \int_{\mathbb{R}} f^* g^*$$

where f^*, g^* denote the symmetric decreasing rearrangements of f, g respectively.

Error-Correcting Codes. Let $k, n \in \mathbb{N}$, and let Σ be a finite alphabet. A *code* is a one-to-one function $C \colon \Sigma^k \to \Sigma^n$ that maps *messages* to *codewords*, where k and n are called the code's *dimension* and *block length*, respectively. The *rate* of the code, measuring the redundancy of the encoding, is defined to be $\rho \stackrel{\text{def}}{=} k/n$. We will sometime identify the code C with its image $C(\Sigma^k)$. In particular, we shall write $c \in C$ to indicate that there exists $x \in \{0,1\}^k$ such that $c = C(x)$, and say that c is a codeword of C. The *relative distance* of a code is the minimal relative distance between two codewords of C, and is denoted by $\delta \stackrel{\text{def}}{=} \min_{c \neq c' \in C} \{\mathrm{dist}(c,c')\}$.

We say that C is an *asymptotically good code* if it has constant rate and constant relative distance. We shall make an extensive use of asymptotically good codes that are *balanced*, that is, codes in which each codeword consists of the same number of 0's and 1's

[9] Note that this is equal, up to a factor 2, to the total variation distance between p and q.

▶ **Proposition 5** (Good Balanced Codes). *For any constant $\delta \in [0,1)$, there exists a good balanced code $C\colon \{0,1\}^k \to \{0,1\}^n$ with relative distance δ and constant rate. Namely, there exists a constant $\rho > 0$ such that the following holds.*

1. *Balance: $|C(x)| = \frac{n}{2}$ for all $x \in \{0,1\}^k$;*
2. *Relative distance: $\mathrm{dist}(C(x), C(y)) > \delta$ for all distinct $x, y \in \{0,1\}^k$;*
3. *Constant rate: $\frac{k}{n} \geq \rho$.*

Proof. Fix any code C' with linear distance δ and constant rate (denoted ρ'). We transform $C'\colon \{0,1\}^k \to \{0,1\}^{n'}$ to a balanced code $C\colon \{0,1\}^k \to \{0,1\}^{2n'}$ by representing 0 and 1 as the balanced strings 01 and 10 (respectively). More accurately, we let $C(x) \stackrel{\text{def}}{=} C'(x) \odot \overline{C'(x)} \in \{0,1\}^n$ for all $x \in \{0,1\}^k$, where $\odot$ denotes the concatenation and $\bar{z}$ is the bitwise negation of z. It is immediate to check that this transformation preserves the distance, and that C is a balanced code with rate $\rho \stackrel{\text{def}}{=} 2\rho'$. ◀

On uniformity. For the sake of notation and clarity, throughout this work we define all algorithms and objects non-uniformly. Namely, we fix the relevant parameter (typically $n \in \mathbb{N}$), and restrict ourselves to inputs or domains of size n (for instance, probability distributions over domain $[n]$). However, we still view it as a generic parameter and allow ourselves to write asymptotic expressions such as $O(n)$. Moreover, although our results are stated in terms of non-uniform algorithms, they can be extended to the uniform setting in a straightforward manner.

4 The Methodology: From Communication Complexity to Distribution Testing

In this section we adapt the methodology for proving property testing lower bounds via reductions from communication complexity, due to Blais, Brody, and Matulef [15], to the setting of distribution testing. As observed in [15, 16], to prove lower bounds on the query complexity of *non-adaptive* testers it suffices to reduce from one-sided communication complexity. We show that for distribution testers (which are inherently non-adaptive), it suffices to reduce from the more restricted communication complexity model of *private-coin* simultaneous message passing (SMP).

Recall that a private-coin SMP protocol for a communication complexity predicate $f\colon \{0,1\}^k \times \{0,1\}^k \to \{0,1\}$ consists of three computationally unbounded parties: Two players (commonly referred to as Alice and Bob), and a Referee. Alice and Bob receive inputs $x, y \in \{0,1\}^k$. Each of the players simultaneously (and independently) sends a message to the referee, based on its input and (private) randomness. The referee is then required to successfully compute $f(x,y)$ with probability at least $2/3$, using its private randomness and the messages received from Alice and Bob. The communication complexity of an SMP protocol is the total number of bits sent by Alice and Bob. The private-coin SMP complexity of f, denoted $\mathsf{SMP}(f)$, is the minimum communication complexity of all SMP protocols that solve f with probability at least $2/3$.

Generally, to reduce an SMP problem f to ε-testing a distribution property Π, Alice and Bob can send messages $m_A(x, r_A, \varepsilon)$ and $m_B(y, r_B, \varepsilon)$ (respectively) to the Referee, where r_A and r_B are the private random strings of Alice and Bob. Subsequently, the Referee uses the messages $m_A(x, r_A, \varepsilon)$ and $m_A(y, r_B, \varepsilon)$, as well as its own private randomness, to feed the property tester samples from a distribution p that satisfies the following conditions: (1) *completeness*: if $f(x,y) = 1$, then $p \in \Pi$; and (2) *soundness*: if $f(x,y) = 0$, then p is ε-far from Π in ℓ_1-distance.

We shall focus on a special type of the foregoing reductions, which is particularly convenient to work with and suffices for all of the our lower bounds. Loosely speaking, in these reductions Alice and Bob both send the prover samples from sub-distributions that can be combined by the Referee to obtain samples from a distribution that satisfies the completeness and soundness conditions. The following lemma gives a framework for proving lower bounds based on such reductions.

▶ **Lemma 6.** *Let $\varepsilon > 0$, and let Ω be a finite domain of cardinality n. Fix a property $\Pi \subseteq \Delta(\Omega)$ and a communication complexity predicate $f\colon \{0,1\}^k \times \{0,1\}^k \to \{0,1\}$. Suppose that there exists a mapping $p\colon \{0,1\}^k \times \{0,1\}^k \to \Delta(\Omega)$ that satisfies the following conditions.*

1. Decomposability: *For every $x, y \in \{0,1\}^k$, there exist constants $\alpha = \alpha(x), \beta = \beta(y) \in [0,1]$ and distributions $p_A(x), p_B(y)$ such that*

$$p(x,y) = \frac{\alpha}{\alpha+\beta} \cdot p_A(x) + \frac{\beta}{\alpha+\beta} \cdot p_B(y)$$

   and α, β can each be encoded with $O(\log n)$ bits.
2. Completeness: *For every $(x,y) = f^{-1}(1)$, it holds that $p(x,y) \in \Pi$.*
3. Soundness: *For every $(x,y) = f^{-1}(0)$, it holds that $p(x,y)$ is ε-far from Π in ℓ_1 distance.*

Then, every ε-tester for Π needs $\Omega\left(\frac{\mathsf{SMP}(f)}{\log(n)}\right)$ samples.

Proof. Supose there exists an ε-tester for Π with sample complexity s'; assume without loss of generality that the soundness of the foregoing tester is $5/6$, at the cost of increasing the query complexity to $s = O(s')$. Let $x, y \in \{0,1\}^k$ be the inputs of Alice and Bob (respectively) for the SMP problem. Alice computes the distribution $p_A(x)$ and the "decomposability parameter" $\alpha = \alpha(x)$ and sends $6s$ independent samples from $p_A(x)$, as well as the parameter α. Analogously, Bob computes $p_B(y)$ and its parameter $\beta = \beta(y)$, and sends $6s$ independent samples from $p_B(y)$ as well as the parameter β. Subsequently, the referee generates a sequence of q independent samples from $p(x,y)$, where each sample is drawn as follows: with probability $\frac{\alpha}{\alpha+\beta}$ use a (fresh) sample from Alice's samples, and with probability $1 - \frac{\alpha}{\alpha+\beta}$ use a (fresh) sample from Bob's samples. Finally the referee feeds the generated samples to the ε-tester for Π.

By Markov's inequality, the above procedure indeed allows the referee to retrieve, with probability at least $1 - \frac{\alpha s}{6s} \geq \frac{5}{6}$, at least s independent samples from the distribution $\frac{\alpha}{\alpha+\beta} \cdot p_A(x) + \frac{\beta}{\alpha+\beta} \cdot p_B(y)$, which equals to $p(x,y)$, by the decomposability condition. If $(x,y) = f^{-1}(1)$, then by the completeness condition $p(x,y) \in \Pi$, and so the ε-tester for Π is successful with probability at least $\frac{5}{6} \cdot \frac{5}{6}$. Similarly, if $(x,y) = f^{-1}(0)$, then by the soundness condition $p(x,y)$ is ε-far from Π, and so the ε-tester for Π is successful with probability at least $\frac{5}{6} \cdot \frac{5}{6}$. Finally, note that since each one of the samples provided by Alice and Bob requires sending $\log n$ bits, the total communication complexity of the protocol is $12s \log n + O(\log n)$ (the last term from the cost of sending α, β), hence $s' = \Omega\left(\frac{\mathsf{SMP}(f)}{\log(n)}\right)$. ◀

We conclude this section by stating a well-known SMP lower bound on the equality problem. Let $\mathrm{EQ}_k\colon \{0,1\}^k \times \{0,1\}^k \to \{0,1\}$ be the equality predicate, i.e., $\mathrm{EQ}_k(x,y) = 1$ if and only if $x = y$. In this work, we shall frequently use the following (tight) lower bound on the EQ_k predicate:

▶ **Theorem 7** (Newman and Szegedy [40])**.** *For every $k \in \mathbb{N}$ it holds that* $\mathsf{SMP}(\mathrm{EQ}_k) = \Omega\left(\sqrt{k}\right)$.

5 The Basic Reduction: The Case of Uniformity

▶ **Theorem 8.** *For any $\varepsilon \in (0, 1/2)$ and finite domain Ω, testing that $p \in \Delta(\Omega)$ is uniform, with respect to proximity parameter ε, requires $\tilde{\Omega}(\sqrt{n})$ samples, where $n = |\Omega|$.*

Proof. Assume there exists a q-query ε-tester for the uniform distribution, with error probability $1/6$. For a sufficiently large $k \in \mathbb{N}$, let $C\colon \{0,1\}^k \to \{0,1\}^n$ be a balanced code as promised by Proposition 5 with distance ε. Namely, there exists an absolute constant $\rho > 0$ such that

1. $|C(x)| = \frac{n}{2}$ for all $x \in \{0,1\}^k$;
2. $\mathrm{dist}(C(x), C(y)) > \varepsilon$ for all distinct $x, y \in \{0,1\}^k$;
3. $\frac{k}{n} \geq \rho$.

Given their respective inputs $x, y \in \{0,1\}^k$ from EQ_k, Alice and Bob separately create inputs $(C(x), C(y)) \in \{0,1\}^n \times \{0,1\}^n$, and the corresponding sets $A \stackrel{\text{def}}{=} \{\, i \in [n] : C(x)_i = 1 \,\}$, $B \stackrel{\text{def}}{=} \{\, i \in [n] : C(y)_i = 0 \,\}$. We then invoke the general reduction of Lemma 6 as follows: we set $\alpha = \beta = \frac{1}{2}$, and $p_A(x) \in \Delta([n])$ (respectively $p_B(y) \in \Delta([n])$) to be the uniform distribution on the set A (respectively B). It is clear that the decomposability condition of the lemma is satisfied for $p(x,y) = \frac{\alpha}{\alpha+\beta} \cdot p_A(x) + \frac{\beta}{\alpha+\beta} \cdot p_B(y) = \frac{1}{2}(p_A(x) + p_B(y))$; we thus turn to the second and third conditions.

Completeness. If $(x,y) \in \mathrm{EQ}_k^{-1}(1)$, then $C(x) = C(y)$ and $A = [n] \setminus B$. This implies that $p(x,y)$ is indeed the uniform distribution on $[n]$, as desired.

Soundness. If $(x,y) \in \mathrm{EQ}_k^{-1}(0)$, then $\mathrm{dist}(C(x), C(y)) > \varepsilon$, and therefore $|A \triangle \bar{B}| > \varepsilon n$ by construction. Since $p(x,y)$ assigns mass $2/n$ to each element in $A \cap B = A \setminus \bar{B}$, and mass 0 to any element in $\bar{A} \cap \bar{B} = \bar{B} \setminus A$, we have $\|p(x,y) - u\|_1 = \frac{1}{n} \cdot |A \triangle \bar{B}| > \varepsilon$; that is, $p(x,y)$ is ε-far from uniform.

The desired $\Omega\left(\frac{\sqrt{n}}{\log n}\right)$ lower bound then immediately follows from Lemma 6 and Theorem 7. ◀

5.1 Obtaining ε-Dependency

In this section, we explain how to generalize the reduction from the previous section to obtain some dependence (albeit non optimal) on the distance parameter ε in the lower bound. This generalization will rely on an extension of the methodology of Lemma 6: instead of having the referee define the distribution $p(x,y)$ as a mixture of $p_A(x)$ and $p_B(y)$ (namely, $p(x,y) = \frac{\alpha(x)}{\alpha(x)+\beta(y)} p_A(x) + \frac{\beta(y)}{\alpha(x)+\beta(y)} p_B(y)$), he will instead use a (random) combination function F_ε, function of ε and its private coins only. Given this function, which maps a larger domain of size $m = \Theta(n/\varepsilon^2)$ to $[n]$, $p(x,y)$ will be defined as the mixture

$$p(x,y) = \frac{\alpha(x)}{\alpha(x)+\beta(y)} p_A(x) \circ F_\varepsilon^{-1} + \frac{\beta(y)}{\alpha(x)+\beta(y)} p_B(y) \circ F_\varepsilon^{-1}.$$

More simply, this allows Alice and Bob to send to the referee samples from their respective distributions on a much larger domain $m \gg n$; the referee, who has on its side chosen how to randomly partition this large domain into only n different "buckets," converts these draws from Alice and Bob into samples from the induced distributions on the n buckets, and takes a mixture of these two distributions instead. By choosing each bucket to chave size roughly $1/\varepsilon^2$, we expect this random "coarsening" of Alice and Bob's distributions to yield a distribution at distance only $\Omega(\varepsilon)$ from uniformity (instead of constant distance) in the no-case; but now letting us get a lower bound on the *original* support size m, i.e. $\tilde{\Omega}\left(\sqrt{n/\varepsilon^2}\right)$, instead of $\tilde{\Omega}(\sqrt{n})$ as before.

▶ **Theorem 9.** *For any $\varepsilon \in (0,1/2)$ and finite domain Ω, testing that $p \in \Delta(\Omega)$ is uniform, with respect to proximity parameter ε, requires $\tilde{\Omega}(\sqrt{n}/\varepsilon)$ samples, where $n = |\Omega|$.*

Proof of Theorem 9. We will reduce from EQ_k, where $k \in \mathbb{N}$ is again assumed big enough (in particular, with regard to $1/\varepsilon^2$). Alice and Bob act as in Section 5, separately creating $(a,b) = (C(x), C(y)) \in \{0,1\}^m \times \in \{0,1\}^m$ from their respective inputs $x, y \in \{0,1\}^k$ (where $C\colon \{0,1\}^k \to \{0,1\}^m$ is a balanced code with linear rate and distance $\delta \stackrel{\text{def}}{=} 1/3$). As before, they consider the sets $A \stackrel{\text{def}}{=} \{\, i \in [m] : C(x)_i = 1 \,\}$, $B \stackrel{\text{def}}{=} \{\, i \in [m] : C(y)_i = 0 \,\}$, set $\alpha = \beta = \frac{1}{2}$, and consider the distributions $p_A(x), p_B(y) \in \Delta([m])$ which are uniform respectively on A and B.

This is where we deviate from the proof of Theorem 8: indeed, setting $n \stackrel{\text{def}}{=} c\varepsilon^2 m$ (where $c > 0$ is an absolute constant determined later), the referee will combine the samples from $p_A(x)$ and $p_B(y)$ in a different way to emulate a distribution $p(x,y) \in \Delta([n])$ – that is, with a much smaller support than that of $p_A(x), p_B(y)$ (instead of setting $p(x,y)$ to be, as before, a mixture of the two).

To do so, the referee randomly partitions $[m]$ into n sets $B_1, \dots, B_n$ of equal size $r \stackrel{\text{def}}{=} |B_j| = \frac{m}{n} = \frac{1}{c\varepsilon^2}$, $j \in [n]$, by choosing a uniformly random equipartition of $[m]$. He then defines the distribution $p = p(x,y) \in \Delta([n])$ by $p(j) = \Pr[\, i \in B_j \,]$ (where $i \in [m]$ is received from either Alice or Bob). Viewed differently, the random equipartition chosen by the referee induces a mapping $F_\varepsilon\colon [m] \to [n]$ such that $\left|F^{-1}(j)\right| = r$ for all $j \in [n]$; and, setting $p'(x,y) = \frac{1}{2}(p_A(x) + p_B(y)) \in \Delta([m])$, we obtain $p(x,y)$ as the *coarsening* of $p'(x,y)$ defined as

$$p(x,y)(j) = \sum_{i \in F_\varepsilon^{-1}(j)} p'(x,y)(i) = p'(x,y)(F_\varepsilon^{-1}(j))$$
$$= \frac{1}{2}\left(p_A(x)(F_\varepsilon^{-1}(j)) + p_B(y)(F_\varepsilon^{-1}(j))\right), \qquad j \in [n].$$

Note furthermore that each sample sent by Alice and Bob (who have no knowledge of the randomly chosen F_ε) can be encoded with $O(\log m) = O(\log \frac{n}{\varepsilon})$ bits.

We then turn to establish the analogue in this generalized reduction of the last two conditions of Lemma 6, i.e. the completeness and soundness. The former, formally stated below, will be an easy consequence of the previous section.

▶ **Claim 10.** *If $x = y$, then $p(x,y)$ is uniform on $[n]$.*

Proof. As in the proof of Theorem 8, in this case the distribution $p'(x,y) = \frac{1}{2}(p_A(x) + p_B(y)) \in \Delta([m])$ is uniform; since each "bucket" $B_j = F_\varepsilon^{-1}(j)$ has the same size, this implies that $p(x,y)(j) = p'(x,y)(B_j) = \frac{1}{n}$ for all $j \in [n]$. ◀

Establishing the soundness, however, is not as straightforward:

▶ **Claim 11.** *If $x \neq y$, then with probability at least $1/100$ (over the choice of the equipartition $(B_1, \dots, B_n)$), $p(x,y)$ is ε-far from uniform.*

Proof. Before delving into the proof, we provide a high-level idea of why this holds. Since the partition was chosen uniformly at random, on expectation each element $j \in [n]$ will have probability $\mathbb{E}[p(x,y)(j)] = \mathbb{E}[p'(x,y)(B_j)] = \frac{1}{n}$. However, since a constant fraction of elements $i \in [m]$ (before the random partition) has probability mass either 0 or $2/m$ (as in the proof of Theorem 8), and each bucket B_j contains $r = 1/(c\varepsilon^2)$ many elements chosen uniformly at random, we expect the fluctuations of $p'(x,y)(B_j)$ around its expectation to be

of the order of $\Omega(\sqrt{r}/m) = \Omega(\varepsilon/n)$ with constant probability, and summing over all j's this will give us the distance $\Omega(\varepsilon)$ we want.

To make this argument precise, we assume $x \neq y$, so that $A \triangle \bar{B} > \delta m$; and define $H \stackrel{\text{def}}{=} A \cap B, L \stackrel{\text{def}}{=} \bar{A} \cap \bar{B}$ (so that $|H| = |L| > \frac{\delta}{2}m$). For any $j \in [n]$, we then let the random variables $H^{(j)}, L^{(j)}$ be the number of "high" and "low" elements of $[m]$ in the bucket B_j, respectively:

$$H^{(j)} \stackrel{\text{def}}{=} |B_j \cap H|\,, \qquad L^{(j)} \stackrel{\text{def}}{=} |B_j \cap L|\,.$$

From the definition, we get that $p = p(x,y)$ satisfies $p(j) = \frac{1}{m}\left(2H^{(j)} + (r - H^{(j)} - L^{(j)}\right) = \frac{r}{m} + \frac{H^{(j)} - L^{(j)}}{m}$ for $j \in [n]$. Furthermore, it is easy to see that $\mathbb{E}[p(j)] = \frac{r}{m} = \frac{1}{n}$ for all $j \in [n]$, where the expectation is over the choice of the equipartition by the referee.

As previously discussed, we will analyze the deviation from this expectation; more precisely, we want to show that with good probability, a constant fraction of the j's will be such that $p(j)$ deviates from $1/n$ by at least an additive $\Omega(\sqrt{r}/m) = \varepsilon/n$. This anticoncentration guarantee will be a consequence of the Paley–Zygmund inequality (Theorem 3) to $Z^{(j)} \stackrel{\text{def}}{=} (H^{(j)} - L^{(j)})^2 \geq 0$; in view of applying it, we need to analyze the first two moments of this random variable.

▶ **Lemma 12.** *For any $j \in [n]$, we have the following. (i) $\mathbb{E}\left[(H^{(j)} - L^{(j)})^2\right] = \delta r \frac{m-r}{m-1}$, and (ii) $\mathbb{E}\left[(H^{(j)} - L^{(j)})^4\right] = 3(1 + o(1))\delta^2 r^2$.*

Proof. Fix any $j \in [n]$. We write for convenience X and Y for respectively $H^{(j)}$ and $L^{(j)}$. The distribution of $(X, Y, r - (X - Y))$ is then a *multivariate hypergeometric distribution* [55] with 3 classes:

$$(X, Y, r - (X + Y)) \sim \text{MultivHypergeom}_3(\underbrace{(\tfrac{1}{2}\delta m, \tfrac{1}{2}\delta m, (1-\delta)m)}_{(K_1, K_2, K_3)}, m, r).$$

Conditioning on $U \stackrel{\text{def}}{=} X + Y$, we have that $\mathbb{E}[X \mid U]$ follows a hypergeometric distribution, specifically $\mathbb{E}[X \mid U] \sim \text{Hypergeom}(U, \frac{1}{2}\delta m, \delta m)$. Moreover, U itself is hypergeometrically distributed, with $U \sim \text{Hypergeom}(r, \delta m, m)$. We can thus write

$$\mathbb{E}\left[(X - Y)^2\right] = \mathbb{E}\left[\mathbb{E}\left[(X - Y)^2 \mid U\right]\right] = \mathbb{E}\left[\mathbb{E}\left[(2X - U)^2 \mid U\right]\right]$$

and

$$\mathbb{E}\left[(X - Y)^4\right] = \mathbb{E}\left[\mathbb{E}\left[(X - Y)^4 \mid U\right]\right] = \mathbb{E}\left[\mathbb{E}\left[(2X - U)^4 \mid U\right]\right].$$

By straightforward, yet tedious, calculations involving the computation of $\mathbb{E}\left[(2X - U)^2 \mid U\right]$ and $\mathbb{E}\left[(2X - U)^4 \mid U\right]$ (after expanding and using the known moments of the hypergeometric

[10] One can also use a formal computation system, e.g. Mathematica:

```
Expectation[ Expectation[(2 X - U)^2, {X \[Distributed] HypergeometricDistribution[U, a*m, 2 a*m]}],
  {U \[Distributed] HypergeometricDistribution[r, 2*a*m, m]}]
Expectation[ Expectation[(2 X - U)^4, {X \[Distributed] HypergeometricDistribution[U, a*m, 2 a*m]}],
  {U \[Distributed] HypergeometricDistribution[r, 2*a*m, m]}]
```

distribution),[10] we obtain

$$\mathbb{E}\big[(X-Y)^2\big] = \delta r \frac{m-r}{m-1} \underset{m\to\infty}{=} (1+o(1))\delta r$$
$$\mathbb{E}\big[(X-Y)^4\big] = \frac{(\delta r(r-m)((-1+3\delta(m-1)-m)m+6r^2(\frac{1}{2}\delta m-1)-6rm(\frac{1}{2}\delta m-1)))}{(m-3)(m-2)(m-1)}$$
$$\xrightarrow[m\to\infty]{} 3\delta^2 r^2 + (1-3\delta)\delta r = 3\delta^2 r^2$$

the last equality as $\delta = 1/3$. ◀

We can now apply the Paley–Zygmund inequality to $Z^{(j)}$. Doing so, we obtain that for $r \le \frac{m}{4}$ (with some slack), and any $\theta \in [0,1]$,

$$\Pr\left[\left|H^{(j)}-L^{(j)}\right| \ge \theta\sqrt{\frac{1}{2}\delta r}\right] \ge \Pr\left[\left|H^{(j)}-L^{(j)}\right| \ge \theta\sqrt{\delta r\frac{m-r}{m-1}}\right]$$
$$\ge (1-\theta^2)^2 \frac{\mathbb{E}\big[(H^{(j)}-L^{(j)})^2\big]^2}{\mathbb{E}\big[(H^{(j)}-L^{(j)})^4\big]}.$$

By the lemma above, the RHS converges to $\frac{(1-\theta^2)^2}{3}$ when $m\to\infty$, and therefore is at least $\frac{(1-\theta^2)^2}{4}$ for m big enough. We set $\theta \stackrel{\text{def}}{=} 1/\sqrt{2}$ to obtain the following: there exists $M \ge 0$ such that

$$\Pr\left[\left|H^{(j)}-L^{(j)}\right| \ge \sqrt{\frac{\delta r}{4}}\right] \ge \frac{1}{16} \tag{4}$$

for every $m \ge M$.

Eq. (4) implies that the number K of *good* indices $j \in [n]$ satisfying $\left|H^{(j)}-L^{(j)}\right| \ge \sqrt{\frac{\delta r}{4}}$ is on expectation at least $\frac{n}{16}$, and by an averaging argument[11] we get that $K \ge \frac{n}{20}$ with probability at least $\frac{1}{76} > \frac{1}{100}$.

Whenever this happens, the distance from p to uniform is at least

$$\sum_{j \text{ good}} \left|p(j)-\frac{1}{n}\right| = \sum_{j \text{ good}} \frac{\left|H^{(j)}-L^{(j)}\right|}{m} \ge \frac{n}{20}\cdot\frac{\sqrt{\frac{\delta r}{4}}}{m} = \frac{\sqrt{\delta r}}{40}\frac{n}{m} = \frac{\sqrt{c}}{40\sqrt{3}}\varepsilon$$

and choosing $c \ge 4800$ so that $\frac{\sqrt{c}}{40\sqrt{3}} \ge 1$ yields the claim. ◀

From this lemma, we can complete the reduction: given a tester $\mathcal{T}$ for uniformity with query complexity q, we first convert it by standard amplification into a tester $\mathcal{T}'$ with failure probability $\delta \stackrel{\text{def}}{=} 1/1000$ and sample complexity $O(q)$. The referee can provide samples from the distribution $p(x,t)$, and on input ε:

- If $x = y$, then $\mathcal{T}'$ will return reject with probability at most $1/200$;
- If $x \ne y$, then $\mathcal{T}'$ will return reject with probability at least $199/200 \cdot 1/100 > 1/200$;

so repeating independently the protocol a constant (fixed in advance) number of times and taking a majority vote enables the referee to solve EQ_k with probability at least $2/3$. Since $\Omega\left(\sqrt{k}\right) = \Omega\left(\sqrt{n/\varepsilon^2}\right)$ bits of communication are required for this, and each sample sent by

[11] Applying Markov's inequality: $\Pr\left[K < \frac{n}{20}\right] = \Pr\left[n-K > \frac{19n}{20}\right] \le \frac{n-\mathbb{E}[K]}{19n/20} \le \frac{15/16}{19/20} = \frac{75}{76}$.

Alice or Bob to the referee only requires $\Theta\left(\log \frac{n}{\varepsilon}\right)$ bits, we get a lower bound of

$$\Omega\left(\frac{\sqrt{n}}{\varepsilon \log \frac{n}{\varepsilon}}\right) = \tilde{\Omega}\left(\frac{\sqrt{n}}{\varepsilon}\right)$$

on the sample complexity of $\mathcal{T}'$, and therefore of $\mathcal{T}$. ◀

6 The K-Functional: An Unexpected Journey

A quantity that will play a major role in our results is the *K-functional between ℓ_1 and ℓ_2*, a specific case of the key operator in interpolation theory introduced by Peetre [43]. We start by recalling below the definition and some of its properties, before establishing (for our particular setting) results that will be crucial to us. (For more on the K-functional and its use in functional analysis, the reader is referred to [12] and [6].)

▶ **Definition 13** (K-functional)**.** Fix any two Banach spaces $(X_0, \|\cdot\|_0), (X_1, \|\cdot\|_1)$. The *$K$-functional* between X_0 and X_1 is the function $K_{X_0,X_1} : (X_0 + X_1) \times (0,\infty) \to [0,\infty)$ defined by

$$K_{X_0,X_1}(x,t) \stackrel{\text{def}}{=} \inf_{\substack{(x_0,x_1)\in X_0\times X_1 \\ x_0+x_1=x}} \|x_0\|_0 + t\|x_1\|_1.$$

For $a \in \ell_1 + \ell_2$, we denote by κ_a the function $t \mapsto K_{\ell_1,\ell_2}(a,t)$.

In other terms, as t varies the quantity $\kappa_a(t)$ interpolates between the ℓ_1 and ℓ_2 norms of the sequence a (and accordingly, for any fixed t it defines a norm on $\ell_1 + \ell_2$). In particular, note that for large values of t the function $\kappa_a(t)$ is close to $\|x\|_1$, whereas for small values of t the function $\kappa_a(t)$ is close to $t\|x\|_2$ (see Corollary 17). We henceforth focus on the case of K_{ℓ_1,ℓ_2}, although some of the results mentioned hold for the general setting of arbitrary Banach X_0, X_1.

▶ **Proposition 14** ([12, Proposition 1.2])**.** *For any $a \in \ell_1 + \ell_2$, κ_a is continuous, increasing, and concave. Moreover, the function $t \in (0,1) \mapsto \frac{\kappa_a}{t}$ is decreasing.*

Although no closed-form expression is known for κ_a, it will be necessary for us to understand its behavior, and therefore seek good upper and lower bounds on its value. We start with the following inequality, due to Holmstedt [34], which, loosely speaking, shows that the infimum in the definition of $\kappa_a(t)$ is *roughly* obtained by partitioning $a = (a_1, a_2)$ such that a_1 consists of heaviest t^2 coordinates of a, and a_2 consists of the rest.

▶ **Proposition 15** ([6, Proposition 2.2], after [34, Theorem 4.2])**.** *For any $a \in \ell_2$ and $t > 0$,*

$$\frac{1}{4}\left(\sum_{i=1}^{\lfloor t^2\rfloor} a_i^* + t\left(\sum_{i=\lfloor t^2\rfloor+1}^{\infty} a_i^{*2}\right)^{\frac{1}{2}}\right) \le \kappa_a(t) \le \sum_{i=1}^{\lfloor t^2\rfloor} a_i^* + t\left(\sum_{i=\lfloor t^2\rfloor+1}^{\infty} a_i^{*2}\right)^{\frac{1}{2}} \tag{5}$$

where a^* is a non-increasing permutation of the sequence $(|a_i|)_{i\in\mathbb{N}}$.

(We remark that for our purposes, this constant factor gap between left-hand and right-hand side is not innocuous, as we will later need to study te behavior of the *inverse* of the function κ_a.)

Incomparable bounds on κ_a were obtained [38], relating it to a different quantity, the "Q-norm," which we discuss and generalize next.

6.1 Approximating the K-Functional by the Q-norm

Loosely speaking, the Q-norm of a vector a (for a given parameter T) is a *mixed* ℓ_1/ℓ_2 norm: it is the maximum one can reach by partitioning the components of a into T sets, and taking the sum of the ℓ_2 norms of these T subvectors. Although not straightforward to interpret, this intuitively captures the notion of *sparsity* of a: indeed, if a is supported on k elements then its Q-norm becomes equal to the ℓ_1 norm for parameter $T \geq k$.

▶ **Proposition 16** ([6, Lemma 2.2], after [38, Lemma 2]). *For arbitrary $a \in \ell_2$ and $t \in \mathbb{N}$, define the norm*

$$\|a\|_{Q(t)} \stackrel{\text{def}}{=} \sup \left\{ \sum_{j=1}^{t} \left(\sum_{i \in A_j} a_i^2 \right)^{1/2} \;:\; (A_j)_{1 \leq j \leq t} \text{ partition of } \mathbb{N} \right\}.$$

Then, for any $a \in \ell_2$, and $t > 0$ such that $t^2 \in \mathbb{N}$, we have

$$\|a\|_{Q(t^2)} \leq \kappa_a(t) \leq \sqrt{2}\|a\|_{Q(t^2)}. \tag{6}$$

As we shall see shortly, one can generalize this result further, obtaining a tradeoff in the upper bound. Before turning to this extension in Lemma 18 and Lemma 21, we first state several other properties of the K-functional implied by the above:

▶ **Corollary 17.** *For any $a \in \ell_2$,*
1. *$\kappa_a(t) = t\|a\|_2$ for all $t \in (0,1)$*
2. *$\lim_{t \to 0^+} \kappa_a(t) = 0$*
3. *$\frac{1}{4}\|a\|_1 \leq \lim_{t \to \infty} \kappa_a(t) \leq \|a\|_1$.*

Moreover, for a supported on finitely many elements, it is the case that $\lim_{t \to \infty} \kappa_a(t) = \|a\|_1$.

Proof. The first two points follow by definition; turning to Corollary 3, we first note the upper bound is a direct consequence of the definition of κ_a as an infimum (as, for all $t > 0$, $\kappa_a(t) \leq \|a\|_1$). (This itself ensures the limit as $t \to \infty$ exists by monotone convergence, as κ_a is a non-decreasing bounded function.) The lower bound follows from that of Theorem 15, which guarantees that for all $t > 0$ $\kappa_a(t) \geq \frac{1}{4}\sum_{i=1}^{\lfloor t^2 \rfloor} a_i^* \xrightarrow[t \to \infty]{} \frac{1}{4}\|a\|_1$. Finally, the last point can be obtained immediately from, e.g., the lower bound side of Proposition 16 and the upper bound given on Corollary 3 above. ◀

▶ **Lemma 18.** *For any $a \in \ell_2$ and t such that $t^2 \in \mathbb{N}$, we have*

$$\|a\|_{Q(t^2)} \leq \kappa_a(t) \leq \|a\|_{Q(2t^2)}. \tag{7}$$

Proof of Lemma 18. We follow and adapt the proof of [6, Lemma 2.2] (itself similar to that of [38, Lemma 2]). The first inequality is immediate: indeed, for any sequence $c \in \ell_2$, by the definition of $\|a\|_{Q(t^2)}$ and the monotonicity of the p-norms, we have $\|c\|_{Q(t^2)} \leq \|c\|_1$; and by Cauchy–Schwarz, for any partition $(A_j)_{1 \leq j \leq t^2}$ of $\mathbb{N}$,

$$\sum_{j=1}^{t^2} \left(\sum_{i \in A_j} c_i^2 \right)^{1/2} \leq t \left(\sum_{j=1}^{t^2} \sum_{i \in A_j} c_i^2 \right)^{1/2} = t\|c\|_2$$

and thus $\|c\|_{Q(t^2)} \leq t\|c\|_2$. This yields the lower bound, as

$$\kappa_a(t) = \inf_{\substack{a'+a''=a \\ a' \in \ell_1, a'' \in \ell_2}} \|a'\|_1 + t\|a''\|_2 \geq \inf_{\substack{a'+a''=a \\ a' \in \ell_1, a'' \in \ell_2}} \|a'\|_{Q(t^2)} + \|a''\|_{Q(t^2)} \geq \|a\|_{Q(t^2)}$$

by the triangle inequality.

We turn to the upper bound. As $\ell_2(\mathbb{R})$ is a symmetric space and $\kappa_a = \kappa_{|a|}$, without loss of generality, we can assume that $(a_k)_{k\in\mathbb{N}}$ is non-negative and monotone non-increasing, i.e. $a_1 \geq a_2 \geq \cdots \geq a_k \geq \ldots$. We will rely on the characterization of κ_a as

$$\kappa_a(t) = \sup\left\{ \sum_{k=1}^{\infty} a_k b_k \;:\; b \in \ell_2, \max(\|b\|_\infty, t^{-1}\|b\|_2) \leq 1 \right\}, \qquad t > 0$$

(see e.g. [6, Lemma 2.2] for a proof). The first step is to establish the existence of a "nice" sequence $b \in \ell_2$ arbitrarily close to this supremum:

▶ **Claim 19.** *For any $\delta > 0$, there exists a* non-increasing, *non-negative sequence $b^* \in \ell_2$ with $\max(\|b^*\|_\infty, t^{-1}\|b^*\|_2) \leq 1$ such that*

$$(1-\delta)\kappa_a \leq \sum_{k=1}^{\infty} a_k b_k^*.$$

Proof. By the above characterization, there exists a sequence $b \in \ell_2$ with $\max(\|b\|_\infty, t^{-1}\|b\|_2) \leq 1$ such that $(1-\delta)\kappa_a \leq \sum_{k=1}^{\infty} a_k b_k$. We now claim that we can further take b to be non-negative and monotone non-increasing as well. The first part is immediate, as replacing negative terms by their absolute values can only increase the sum (since a is itself non-negative). For the second part, we will invoke the Hardy–Littlewood rearrangement inequality (Theorem 4), which states that for any two non-negative functions f, g vanishing at infinity, the integral $\int_{\mathbb{R}} fg$ is maximized when f and g are non-increasing. We now apply this inequality to a, b, letting a^*, b^* be the non-increasing rearrangements of a, b (in particular, we have $a = a^*$) and introducing the functions f_a, f_b:

$$f_a = \sum_{j=1}^{\infty} a_j \mathbb{1}_{(j-1,j]}, \quad f_b = \sum_{j=1}^{\infty} b_j \mathbb{1}_{(j-1,j]}$$

which satisfy the hypotheses of Theorem 4. Thus, we get $\int_{\mathbb{R}} f_a f_b \leq \int_{\mathbb{R}} f_a^* f_b^*$; as it is easily seen that $f_a^* = f_{a^*}$ and $f_b^* = f_{b^*}$, this yields

$$\sum_{k=1}^{\infty} a_k b_k = \int_{\mathbb{R}} f_a f_b \leq \int_{\mathbb{R}} f_a^* f_b^* = \sum_{k=1}^{\infty} a_k^* b_k^* = \sum_{k=1}^{\infty} a_k b_k^*.$$

Moreover, it is immediate to check that $\max(\|b^*\|_\infty, t^{-1}\|b^*\|_2) \leq 1$. ◀

The next step is to relate the inner product $\sum_{k=1}^{\infty} a_k b_k^*$ to the Q-norm of a:

▶ **Claim 20.** *Fix $t > 0$ such that $t^2 \in \mathbb{N}$, and let $b^* \in \ell_2$ be any non-increasing, non-negative sequence with $\max(\|b^*\|_\infty, t^{-1}\|b^*\|_2) \leq 1$. Then*

$$\sum_{k=1}^{\infty} a_k b_k^* \leq \|a\|_{Q(2t^2)}.$$

Proof. We proceed constructively, by exhibiting a partition of $\mathbb{N}$ into $2t^2$ sets $A_1, \ldots, A_{2t^2}$ satisfying $\sum_{k=1}^{\infty} a_k b_k^* \leq \sum_{j=1}^{2t^2} \left(\sum_{i\in A_j} {b_i^*}^2\right)^{1/2}$. This will prove the claim, by definition of $\|a\|_{Q(2t^2)}$ as the supremum over all such partitions.

Specifically, we inductively choose $n_0, n_1, \ldots, n_T \in \{0, \ldots, \infty\}$ as follows, where $T \stackrel{\text{def}}{=} \frac{t^2}{c}$ for some $c > 0$ to be chosen later (satisfying $T \in \mathbb{N}$). If $0 = n_0 < n_1 < \cdots < n_m$ are already

set, then

$$n_{m+1} \stackrel{\text{def}}{=} 1 + \sup\left\{ \ell \geq n_m \ : \ \sum_{i=n_m+1}^{\ell} {b_i^*}^2 \leq c \right\}.$$

From $\|b^*\|_2 \leq t$, it follows that $n_T = \infty$. Let m^* be the first index such that $n_{m^*+1} > n_{m^*}+1$. Note that this implies (by monotonicity of b^*) that ${b_i^*}^2 > c$ for all $i \leq n_{m^*}$, and ${b_i^*}^2 \leq c$ for all $i \geq n_{m^*}+1$. We can write

$$\sum_{i=1}^{\infty} a_i b_i^* = \sum_{m=1}^{T} \sum_{i=n_{m-1}+1}^{n_m} a_i b_i^* = \sum_{i=1}^{n_{m^*}} a_i b_i^* + \sum_{m=m^*+1}^{T} \sum_{i=n_{m-1}+1}^{n_m} a_i b_i^*$$

Since $\|b^*\|_\infty \leq 1$ and $n_{m-1}+1 = n_m$ for all $m \leq m^*$, the first term can be bounded as

$$\sum_{i=1}^{n_{m^*}} a_i b_i^* \leq \sum_{i=1}^{n_{m^*}} \sqrt{a_i^2} = \sum_{m=1}^{m^*} \left(\sum_{i=n_{m-1}+1}^{n_m} a_i^2 \right)^{1/2}.$$

Turning to the second term, we recall that ${b_i^*}^2 \leq c$ for all $i \geq n_{m^*}+1$, so that $\sum_{i=n_{m-1}+1}^{n_m} {b_i^*}^2 \leq 2c$ for all $m \geq m^*+1$. This allows us to bound the second term as

$$\begin{aligned}\sum_{m=m^*+1}^{T} \sum_{i=n_{m-1}+1}^{n_m} a_i b_i^* &\leq \sum_{m=m^*+1}^{T} \left(\sum_{i=n_{m-1}+1}^{n_m} {b_i^*}^2 \right)^{1/2} \left(\sum_{i=n_{m-1}+1}^{n_m} a_i^2 \right)^{1/2} \\ &\leq \sqrt{2c} \sum_{m=m^*+1}^{T} \left(\sum_{i=n_{m-1}+1}^{n_m} a_i^2 \right)^{1/2}\end{aligned}$$

Therefore, by combining the two we get that

$$\begin{aligned}(1-\delta)\kappa_a(t) &\leq \sum_{m=1}^{m^*} \left(\sum_{i=n_{m-1}+1}^{n_m} a_i^2 \right)^{1/2} + \sqrt{2c} \sum_{m=m^*+1}^{T} \left(\sum_{i=n_{m-1}+1}^{n_m} a_i^2 \right)^{1/2} \\ &\leq \max(1, \sqrt{2c}) \sum_{m=1}^{T} \left(\sum_{i=n_{m-1}+1}^{n_m} a_i^2 \right)^{1/2} \leq \max(1, \sqrt{2c}) \|a\|_{Q(T)} = \|a\|_{Q(2t^2)}\end{aligned}$$

the last equality by choosing $c \stackrel{\text{def}}{=} \frac{1}{2}$. ◀

We now fix an arbitrary $\delta > 0$, and let b^* be as promised by Claim 19. As this sequence satisfies the assumptions of Claim 20, putting the two results together leads to

$$(1-\delta)\kappa_a(t) \leq \sum_{k=1}^{\infty} a_k b_k^* \leq \|a\|_{Q(2t^2)}.$$

Since this holds for all $\delta > 0$, taking the limit as $\delta \searrow 0$ gives the (upper bound of the) lemma. ◀

We observe that, with similar techniques, one can also establish the following generalization of Proposition 16:

▶ **Lemma 21** (Generalization of Proposition 16). *For any $a \in \ell_2$, t, and $\alpha \in [1, \infty)$ such that $t^2, \alpha t^2 \in \mathbb{N}$, we have*

$$\|a\|_{Q(t^2)} \leq \kappa_a(t) \leq \sqrt{1+\alpha^{-1}}\|a\|_{Q(\alpha t^2)}. \tag{8}$$

Proof of Lemma 21 (Sketch). We again follow the proof of [6, Lemma 2.2], up to the inductive definition of $n_1, \ldots, n_j$, which we change as

$$n_{m+1} = 1 + \sup\left\{ \ell \geq n_m : \sum_{i=n_m+1}^{\ell} b_i^2 \leq \frac{1}{\alpha} \right\}.$$

Since $\|b\|_\infty \leq 1$, we have $\sum_{i=n_m+1}^{n_{m+1}} b_i^2 \leq 1 + \frac{1}{\alpha}$. From $\|b\|_2 \leq t$, it follows that $n_{\alpha t^2} = \infty$. Therefore, for any $\delta > 0$,

$$(1-\delta)\kappa_a(t) \leq \sum_{i=1}^{\infty} a_i b_i \leq \sum_{m=1}^{T} \left(\sum_{i=n_{m-1}+1}^{n_m} b_i^2 \right)^{1/2} \left(\sum_{i=n_{m-1}+1}^{n_m} a_i^2 \right)^{1/2} \leq \sqrt{1+\frac{1}{\alpha}}\|a\|_{Q(\alpha t^2)}.$$

Since this holds for all $\delta > 0$, taking the limit gives the (upper bound of the) lemma. ◀

We note that further inequalities relating κ_a to other functionals of a were obtained in [33].

6.2 Concentration Inequalities for Weighted Rademacher Sums

The connection between the K-functional and tail bounds on weighted sums of Rademacher random variables was first made by Montgomery-Smith [38], to which the following result is due (we here state a version with slightly improved constants):

▶ **Theorem 22.** *Let $(X_i)_{i\in\mathbb{N}}$ be a sequence of independent Rademacher random variables, i.e. uniform on $\{-1, 1\}$. Then, for any $a \in \ell_2$ and $t > 0$,*

$$\Pr\left[\sum_{i=1}^{\infty} a_i X_i \geq \kappa_a(t) \right] \leq e^{-\frac{t^2}{2}}. \tag{9}$$

and, for any fixed $c > 0$ and all $t \geq 1$,

$$\Pr\left[\sum_{i=1}^{\infty} a_i X_i \geq \frac{1}{1+c}\kappa_a(t) \right] \geq e^{-\left(\frac{2}{c}\ln\frac{\sqrt{6}(1+c)}{c}\right)(t^2+c)}. \tag{10}$$

In particular,

$$\Pr\left[\sum_{i=1}^{\infty} a_i X_i \geq \frac{1}{2}\kappa_a(t) \right] \geq e^{-(\ln 24)(t^2+1)} \geq e^{-(2\ln 24)t^2}.$$

One can intepret the above theorem as stating that the (inverse of the) K-functional κ_a is the "right" parameter to consider in these tail bounds; while standard Chernoff or Hoeffding bounds will depend instead on the quantity $\|a\|_2$. Before giving the proof of this theorem, we remark that similar statements or improvements can be found in [33] and [6]; below, we closely follow the argument of the latter.

Proof of Theorem 22. The upper bound can be found in e.g. [38], or [6, Theorem 2.2]. For the lower bound, we mimic the proof due to Astashkin, improving the parameters of some of the lemmas it relies on.

▶ **Lemma 23** (Small improvement of (2.14) in [6, Lemma 2.3]). *If $a = (a_k)_{k\geq 1} \in \ell_2$, then, for any $\lambda \in (0,1)$,*

$$\Pr\left[\left|\sum_{k=1}^{\infty} a_k X_k\right|^2 \geq \lambda \sum_{k=1}^{\infty} a_k^2\right] \geq \frac{1}{3}(1-\lambda)^2. \tag{11}$$

Proof of Lemma 23. The proof is exactly the same, but when invoking (1.10) for $p = 4$ we use the actual tight version proven there for $p = 2m$ (instead of the more general version that also applies to odd values of p): since $m = 2$, we get $\frac{(2m)!}{2^m m!} = 3$, and $\mathbb{E}[f]^2 \geq \frac{1}{3}\mathbb{E}[f^2]$ in the proof (instead of $(\frac{p}{2}+1)^{-\frac{p}{2}} = \frac{1}{9}$). ◀

Using the lemma above along with Lemma 18 in the proof of [6, Theorem 2.2], we can strenghten it as follows: letting $T \stackrel{\text{def}}{=} \frac{t^2}{c}$, for arbitrary $\delta > 0$ we fix a partition $A_1, \ldots, A_T$ of $\mathbb{N}$ such that $\|a\|_{Q(T)} \leq (1+\delta)\sum_{j=1}^{T}\left(\sum_{k\in A_j} a_k^2\right)^{1/2}$,

$$\begin{aligned}
\Pr\left[\sum_{k=1}^{\infty} a_k X_k > \frac{1}{1+c}\kappa_a(t)\right] &\geq \Pr\left[\sum_{k=1}^{\infty} a_k X_k > \frac{1}{\sqrt{1+c}}\|a\|_{Q(T)}\right] && \text{(by (7))}\\
&\geq \Pr\left[\sum_{j=1}^{T}\sum_{k\in A_j} a_k X_k > \frac{1+\delta}{\sqrt{1+c}}\sum_{j=1}^{T}\left(\sum_{k\in A_j} a_k^2\right)^{1/2}\right]\\
&\geq \prod_{j=1}^{T}\Pr\left[\sum_{k\in A_j} a_k X_k > \frac{1+\delta}{\sqrt{1+c}}\left(\sum_{k\in A_j} a_k^2\right)^{1/2}\right]\\
&= \prod_{j=1}^{T}\frac{1}{2}\Pr\left[\left|\sum_{k\in A_j} a_k X_k\right|^2 > \left(\frac{1+\delta}{\sqrt{1+c}}\right)^2\left(\sum_{k\in A_j} a_k^2\right)\right]\\
& && \text{(symmetry)}\\
&\geq \prod_{j=1}^{T}\frac{1}{6}\left(1-\frac{(1+\delta)^2}{1+c}\right)^2. && \text{(Lemma 23)}
\end{aligned}$$

By taking the limit as $\delta \to 0^+$, we then obtain

$$\begin{aligned}
\Pr\left[\sum_{k=1}^{\infty} a_k X_k > \frac{1}{1+c}\kappa_a(t)\right] &\geq \left(\frac{1}{6}\left(1-\frac{1}{1+c}\right)^2\right)^T\\
&= \left(\frac{c}{\sqrt{6}(1+c)}\right)^{\frac{2t^2}{c}} = e^{-\left(\frac{2}{c}\ln\frac{\sqrt{6}(1+c)}{c}\right)t^2}. \qquad (12)
\end{aligned}$$

This takes care of the case where $\frac{t^2}{c}$ is an integer. If this is not the case, we consider $s \stackrel{\text{def}}{=} \sqrt{c\left(\lfloor\frac{t^2}{c}\rfloor + 1\right)}$, so that $t^2 \leq s^2 \leq t^2 + c$. The monotonicity of κ_a then ensures that

$$\begin{aligned}
\Pr\left[\sum_{k=1}^{\infty} a_k X_k > \frac{1}{1+c}\kappa_a(t)\right] &\geq \Pr\left[\sum_{k=1}^{\infty} a_k X_k > \frac{1}{1+c}\kappa_a(s)\right]\\
&\underset{(12)}{\geq} e^{-\left(\frac{2}{c}\ln\frac{\sqrt{6}(1+c)}{c}\right)s^2} \geq e^{-\left(\frac{2}{c}\ln\frac{\sqrt{6}(1+c)}{c}\right)(t^2+c)}
\end{aligned}$$

which concludes the proof. ◀

6.3 Some Examples

To gain intuition about the behavior of κ_a, we now compute tight asymptotic expressions for it in several instructive cases, specifically for some natural examples of probability distributions in $\Delta([n])$.

From the lower bound of Proposition 16 and the fact that $\kappa_p \leq \|p\|_1$ for any $p \in \ell_1$, it is clear that as soon as $t \geq \sqrt{n}$, $\kappa_p(t) = 1$ for any $p \in \Delta([n])$. It suffices then to consider the case $0 \leq t \leq \sqrt{n}$.

The uniform distribution. We let p be the uniform distribution on $[n]$: $p_k = \frac{1}{n}$ for all $i \in [n]$. By considering a partition of $[n]$ into t^2 sets of size $\frac{n}{t^2}$, the lower bound of Proposition 16 yields $\kappa_p(t) \geq \|p\|_{Q(t^2)} \geq \frac{t}{\sqrt{n}}$. On the other hand, by definition $\kappa_p(t) = \inf_{p'+p''=p} \|p'\|_1 + t\|p''\|_2 \leq t\|p\|_2 = \frac{t}{\sqrt{n}}$, and thus

$$\kappa_p(t) = \begin{cases} \frac{t}{\sqrt{n}} & \text{if } t \leq \sqrt{n} \\ 1 & \text{if } t \geq \sqrt{n}. \end{cases}$$

We remark that in this case, the upper bound of Holmstedt from Proposition 15 only results in

$$\kappa_p(t) \leq \frac{t^2}{n} + t\sqrt{\frac{n-t^2}{n^2}} = f\left(\frac{t}{\sqrt{n}}\right)$$

where $f\colon x \in [0,1] \mapsto x^2 + x\sqrt{1-x^2}$. It is instructive to note this shows that this could not possibly have been the right upper bound (and therefore that Theorem 15 cannot be tight in general), as f is neither concave nor non-decreasing, and not even bounded by 1:

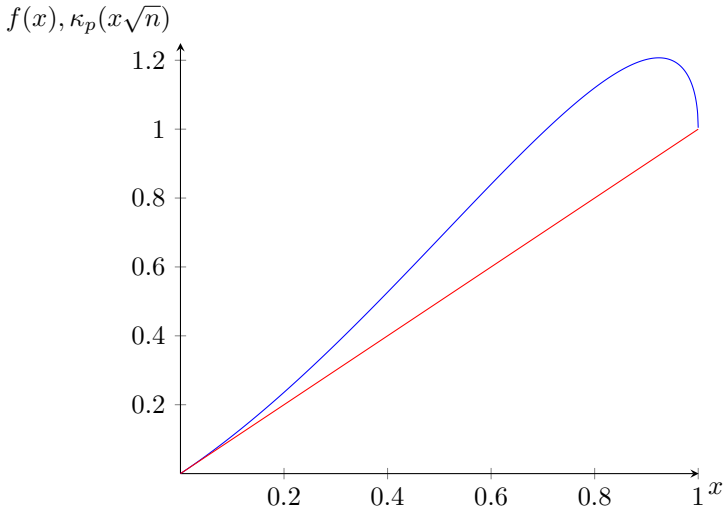


Figure 2 Holmstedt's upper bound (in blue) vs. true behavior of κ_p (in red).

From the above, we can now compare the behavior of $\kappa_p^{-1}(1-2\varepsilon)$ to the "2/3-norm functional" introduced by Valiant and Valiant [53]: for $\varepsilon \in (0, 1/2)$,

$$\kappa_p^{-1}(1-2\varepsilon) = (1-2\varepsilon)\sqrt{n}, \qquad \|p_{-\varepsilon}^{-\max}\|_{2/3} = (1-\varepsilon)^{3/2}\sqrt{n} + o(1). \tag{13}$$

The Harmonic distribution. We now consider the case of the (truncated) Harmonic distribution, letting $p \in \Delta([n])$ be defined as $p_k = \frac{1}{kH_n}$ for all $i \in [n]$ (H_n being the n-th Harmonic number). By considering a partition of $[n]$ into $t^2 - 1$ sets of size 1 and one of size $n - t^2$, the lower bound of Proposition 16 yields

$$H_n \kappa_p(t) \geq \|p\|_{Q(t^2)} \geq \sum_{k=1}^{t^2-1} \frac{1}{k} + \sqrt{\sum_{k=t^2}^{n} \frac{1}{k^2}}$$

while Holmstedt's upper bound gives

$$H_n \kappa_p(t) \leq \sum_{k=1}^{t^2-1} \frac{1}{k} + t\sqrt{\sum_{k=t^2}^{n} \frac{1}{k^2}}.$$

For $t = O(1)$, this implies that $\kappa_p(t) = o(1)$; however, for $t = \omega(1)$ (but still less than $\sqrt{n}$), an asymptotic development of both upper and lower bounds shows that

$$\kappa_p(t) = \frac{2 \ln t + O(1)}{\ln n}.$$

Using this expression, we can again compare the behavior of $\kappa_p^{-1}(1 - 2\varepsilon)$ to the 2/3-norm functional of [53]: for $\varepsilon \in (0, 1/2)$,

$$\kappa_p^{-1}(1 - 2\varepsilon) = \Theta\left(n^{\frac{1}{2} - \varepsilon}\right), \qquad \|p_{-\varepsilon}^{-\max}\|_{2/3} = \Theta\left(\frac{n^{\frac{1-\varepsilon}{2}}}{\log n}\right) = \Theta\left(n^{\frac{1-\varepsilon}{2} - o(1)}\right). \tag{14}$$

7 Identity Testing, revisited

For any $x \in (0, 1/2)$ and sequence $a \in \ell_1$, we let $t_x \stackrel{\text{def}}{=} \kappa_a^{-1}(1 - 2x)$, where κ_a is the K-functional of a as previously defined. Armed with the results and characterizations from the previous section, we will first in Section 7.1 describe an elegant reduction from communication complexity leading to a lower bound on instance-optimal identity testing parameterized by the quantity t_ε. Guided by this lower bound, we then will in Section 7.2 consider this result from the *upper bound* viewpoint, and in Theorem 30 establish that indeed this parameter captures the sample complexity of this problem. Finally, Section 7.3 is concerned with tightening our lower bound by using different arguments: specifically, showing that the bound that appeared naturally as a consequence of our communication complexity approach can, in hindsight, be established and slightly strenghtened with standard distribution testing arguments.

7.1 The Communication Complexity Lower Bound

In this subsection we prove the following lower bound on identity testing, via reduction from SMP communication complexity.

▶ **Theorem 24.** *Let Ω be a finite domain, and let $p = (p_1, \ldots, p_n) \in \Delta(\Omega)$ be a distribution, given as a parameter. Let $\varepsilon \in (0, 1/5)$, and set $t_\varepsilon \stackrel{\text{def}}{=} \kappa_p^{-1}(1 - 2\varepsilon)$. Then, given sample access to a distribution $q = (q_1, \ldots, q_n) \in \Delta(\Omega)$, testing $p = q$ versus $\|p - q\|_1 > \varepsilon$ requires $\Omega(t_\varepsilon / \log(n))$ samples from q.*

We will follow the argument outlined in Section 2.2: namely, applying the same overall idea as in the reduction for uniformity testing, but with an error-correcting code specifically designed for the distribution p instead of a standard Hamming one. To prove Theorem 24 we thus first need to define and obtain codes with properties that are tailored for our reduction; which we do next.

7.1.1 Balanced p-weighted codes

Recall that in our reductions so far, the first step is for Alice and Bob to apply a code to their inputs; typically, we chose that code to be a balanced code with constant rate, and linear distance *with respect to the uniform distribution* (i.e., with good Hamming distance). In order to obtain better bounds on a case-by-case basis, it will be useful to consider a generalization of these codes, under a different distribution:

▶ **Definition 25** (p-distance)**.** For any $n \in \mathbb{N}$, given a probability distribution $p \in \Delta([n])$ we define the *p-distance* on $\{0,1\}^n$, denoted dist_p, as the weighted Hamming distance

$$\mathrm{dist}_p(x,y) \stackrel{\text{def}}{=} \sum_{i=1}^{n} p(i) \cdot |x_i - y_i|$$

for $x, y \in \{0,1\}^n$. (In particular, this is a pseudometric on $\{0,1\}^n$.) The *p-weight* of $x \in \{0,1\}^n$ is given by $\mathrm{weight}_p(x) \stackrel{\text{def}}{=} \mathrm{dist}_p(x, 0^n)$.

A p-weighted code is a code whose distance guarantee is with respect to the p-distance.

▶ **Definition 26** (p-weighted codes)**.** Fix a probability distribution $p \in \Delta([n])$. We say that $C\colon \{0,1\}^k \to \{0,1\}^n$ is a (binary) *p-weighted code with relative distance $\gamma = \gamma(n)$ and rate $\rho = k/n$* if

$$\mathrm{dist}_p(C(x), C(y)) > \gamma$$

for all distinct $x, y \in \{0,1\}^k$.

Recall that the "vanilla" reduction in Section 5 relies on *balanced* codes. We generalize the balance property to the p-distance and allow the following relaxation.

▶ **Definition 27** (p-weighted τ-balance)**.** A p-weighted code $C\colon \{0,1\}^k \to \{0,1\}^n$ is *τ-balanced* if there exists $\tau \in (0,1)$ such that $\mathrm{weight}_p(C(x)) \in \left(\frac{1}{2} - \tau, \frac{1}{2} + \tau\right)$ for all $x \in \{0,1\}^k$.

Now, for a distribution p, the volume of the p-ball in $\{0,1\}^n$ is given by

$$\mathrm{Vol}_{\mathbb{F}_2^n, \mathrm{dist}_p}(\varepsilon) \stackrel{\text{def}}{=} \left|\left\{\ w \in \mathbb{F}_2^n\ :\ \ \mathrm{weight}_p(w) \le \varepsilon\ \right\}\right|.$$

Next, we show that there exist nearly balanced p-weighted codes with constant relative distance nearly optimal rate.

▶ **Proposition 28** (Existence of nearly balanced p-weighted codes)**.** *Fix a probability distribution $p \in \Delta([n])$, constants $\gamma, \tau \in (0, \frac{1}{3})$, and $\varepsilon = \max\{\gamma, \frac{1}{2} - \tau\}$. There exists a p-weighted τ-balanced code $C\colon \{0,1\}^k \to \{0,1\}^n$ with relative distance γ such that $k = \Omega(n - \log \mathrm{Vol}_{\mathbb{F}_2^n, \mathrm{dist}_p}(\varepsilon))$.*

In contrast, by the sphere packing bound, every p-weighted code $C\colon \{0,1\}^k \to \{0,1\}^n$ with distance γ satisfies

$$\underbrace{2^k}_{\#\text{codewords}} \le \frac{2^n}{\mathrm{Vol}_{\mathbb{F}_2^n, \mathrm{dist}_p}(\gamma/2)}.$$

Hence, we have $k \le n - \log \mathrm{Vol}_{\mathbb{F}_2^n, \mathrm{dist}_p}(\gamma/2)$.

Proof of Theorem 28. Note that

$$\mathrm{Vol}_{\mathbb{F}_2^n,\mathrm{dist}_p}(\varepsilon) = \left|\left\{\, w \in \mathbb{F}_2^n \,:\, \mathrm{weight}_p(w) \le \varepsilon \,\right\}\right| = 2^n \cdot \Pr_{w\sim\{0,1\}^n}\left[\sum_{i=1}^n p_i w_i \le \varepsilon\right].$$

The probability that a randomly chosen code $C\colon \{0,1\}^k \to \{0,1\}^n$ does *not* have distance γ is

$$\begin{aligned}\Pr_C\left[\,\exists x,y \in \{0,1\}^k \text{ such that } \mathrm{dist}_p(C(x),C(y)) \le \gamma\,\right] &\le 2^{2k} \cdot \Pr_{w,w'\sim\{0,1\}^n}\left[\,\mathrm{dist}_p(w,w') \le \gamma\,\right] \\ &\le 2^{2k} \cdot \Pr_{w\sim\{0,1\}^n}\left[\sum_{i=1}^n p_i w_i \le \varepsilon\right] \\ &= \frac{\mathrm{Vol}_{\mathbb{F}_2^n,\mathrm{dist}_p}(\varepsilon)}{2^{n-2k}}.\end{aligned}$$

Hence, for sufficiently small $k = \Omega(n - \log \mathrm{Vol}_{\mathbb{F}_2^n,\mathrm{dist}_p}(\varepsilon))$, the probability that a random code is a p-weighted code with relative distance γ is at least $2/3$; fix such k. Similarly, the probability that a random code $C\colon \{0,1\}^k \to \{0,1\}^n$ is not τ-balanced (under the p-distance) is

$$\begin{aligned}&\Pr_C\left[\exists x \in \{0,1\}^k \text{ such that } \mathrm{weight}_p(C(x)) \notin \left(\frac{1}{2}-\tau, \frac{1}{2}+\tau\right)\right] \\ &\le 2^k \cdot \Pr_{w\in\{0,1\}^n}\left[\left|\mathrm{weight}_p(w) - \frac{1}{2}\right| > \tau\right] \\ &\le 2^{k+1} \cdot \Pr_{w\in\{0,1\}^n}\left[\sum_{i=1}^n p_i w_i < \varepsilon\right] \\ &\le \frac{\mathrm{Vol}_{\mathbb{F}_2^n,\mathrm{dist}_p}(\varepsilon)}{2^{n-k-1}}.\end{aligned}$$

Thus, the probability that a random code is τ-balanced (under the p-distance) is at least $2/3$, and so, with probability at least $\frac{1}{3}$, a random code satisfies the proposition's hypothesis. ◀

We now establish a connection between the rate of p-weighted codes and the K-functional of p, as introduced in Section 6:

▶ **Claim 29.** *Let $p \in \Delta([n])$ be a probability distribution. Then, for any $\gamma \in (0, \frac{1}{2})$ we have*

$$n - \log \mathrm{Vol}_{\mathbb{F}_2^n,\mathrm{dist}_p}(\gamma) \ge \frac{1}{2\ln 2}\kappa_p^{-1}(1-2\gamma)^2$$

where $\kappa_p^{-1}(u) = \inf\left\{\, t \in (0,\infty) \,:\, \kappa_a(t) \ge u \,\right\}$ for $u \in [0,\infty)$.

Proof. From the definition,

$$\begin{aligned}\mathrm{Vol}_{\mathbb{F}_2^n,\mathrm{dist}_p}(\gamma) = \left|\left\{\, w \in \mathbb{F}_2^n \,:\, \mathrm{weight}_p(w) \le \gamma \,\right\}\right| &= \left|\left\{\, w \in \mathbb{F}_2^n \,:\, \sum_{i=1}^n p_i w_i \le \gamma \,\right\}\right| \\ &= 2^n \Pr_{Y\sim\{0,1\}^n}\left[\sum_{i=1}^n p_i Y_i \le \gamma\right] = 2^n \Pr_{X\sim\{-1,1\}^n}\left[\sum_{i=1}^n p_i X_i \ge 1-2\gamma\right] \\ &= 2^n \Pr_{X\sim\{-1,1\}^n}\left[\sum_{i=1}^n p_i X_i \ge \kappa_p(u_\gamma)\right]\end{aligned}$$

where we set $u_\gamma \stackrel{\text{def}}{=} \kappa_p^{-1}(1-2\gamma)$. From Theorem 22, we then get $\mathrm{Vol}_{\mathbb{F}_2^n,\mathrm{dist}_p}(\gamma) \le 2^n e^{-\frac{u_\gamma^2}{2}}$, from which

$$n - \log \mathrm{Vol}_{\mathbb{F}_2^n,\mathrm{dist}_p}(\gamma) \ge -\log e^{-\frac{u_\gamma^2}{2}} = \frac{1}{2\ln 2} u_\gamma^2$$

as claimed. ◀

7.1.2 The Reduction

Equipped with the nearly balanced p-weighted codes in Theorem 28, we are ready to prove Theorem 24. Assume there exists an s-sample ε-tester for identity to p, with error probability $1/6$, and assume, without loss of generality, that ε is a constant (independent of n).

Fix $\gamma = \varepsilon$ and $\tau = (1-2\varepsilon)/2$. For a sufficiently large $k \in \mathbb{N}$, let $C\colon \{0,1\}^k \to \{0,1\}^n$ be a τ-balanced p-weighted code with relative distance γ, as guaranteed by Theorem 28; namely, the code C satisfies the following conditions.

1. *Balance*: $\mathrm{weight}_p(C(x)) \in \left(\frac{1}{2}-\tau, \frac{1}{2}+\tau\right)$ for all $x \in \{0,1\}^k$;
2. *Distance*: $\mathrm{dist}_p(C(x), C(y)) > \gamma$ for all distinct $x, y \in \{0,1\}^k$;
3. *Rate*: $k = \Omega(n - \log \mathrm{Vol}_{\mathbb{F}_2^n,\mathrm{dist}_p}(\varepsilon))$.

We reduce from the problem of equality in the (private coin) SMP model. Given their respective inputs $x, y \in \{0,1\}^k \times \in \{0,1\}^k$ from EQ_k, Alice and Bob separately create inputs $(a,b) = (C(x), C(y)) \in \{0,1\}^n \times \in \{0,1\}^n$. Let $A \subseteq [n]$ denote the set indicated by a, and let $B \subseteq [n]$ denote the set indicated by b. Alice and Bob then each send to the referee the p-weight of their encoded input, $\mathrm{weight}_p(a) = p(A)$ and $\mathrm{weight}_p(b) = p(B)$ respectively,[12] as well as a sequence of $6cs$ samples independently drawn from the distribution p *restricted* to the subsets A and B respectively, where c is the constant such that $\frac{1}{c}p(B)\cdot \le p(A) \le c \cdot p(B)$, guaranteed by the balance property of C. Finally, the referee checks that $p(A) + p(B) = 1$ (and otherwise rejects) and generates a sequence of q samples by choosing independently, for each of them, Alice's element with probability $p(A)$ and Bob's with probability $p(B)$, and feeds these samples to the ε-tester for identity to p.

By Markov's inequality, the above procedure indeed allows the referee to retrieve, with probability at least $1 - \frac{cs}{6cs} = \frac{5}{6}$, at least s independent samples from the distribution

$$q \stackrel{\text{def}}{=} p(A)\cdot p|_A + p(B)\cdot p|_B,$$

at the cost of $O(s \log n)$ bits of communication in total.

For correctness, note that if $x = y$, then $A = \bar{B}$, which implies $q = p$. On the other hand, if $x \neq y$, by the (p-weighted) distance of C we have $\mathrm{dist}_p(C(x), C(y)) > \gamma$, and so $p(A \cap B) + p(\overline{A \cup B}) > \gamma$. Note that every $i \in A \cap B$ satisfies $q_i = 2p_i$ and every $i \in \overline{A \cup B}$ is not supported in q. Therefore, we have $\|p - q\|_1 > \varepsilon$. The referee can therefore invoke the identity testing algorithm to distinguish between p and q with probability $1-(\frac{1}{6}+\frac{1}{6}) = \frac{2}{3}$. This implies that the number of samples q used by any such tester must satisfy $s \log n = \Omega\left(\sqrt{k}\right)$. Finally, by Claim 29 we have

$$k = \Omega\big(n - \log \mathrm{Vol}_{\mathbb{F}_2^n,\mathrm{dist}_p}(\varepsilon)\big) = \Omega\big(\kappa_p^{-1}(1-2\varepsilon)^2\big),$$

and therefore we obtain a lower bound of $s = \Omega(t_\varepsilon / \log(n))$.

[12] A standard argument shows it suffices to specify $p(A)$ and $p(B)$ with precision roughly $1/n^2$, and so sending the weights only costs $O(\log n)$ bits.

7.2 The Upper Bound

Inspired by the results of the previous section, it is natural to wonder whether the dependence on t_ε of the lower bound is the "right" one. Our next theorem shows that this is the case: the parameter t_ε does, in fact, capture the sample complexity of the problem.

▶ **Theorem 30.** *There exists an absolute constant $c > 0$ such that the following holds. Given any fixed distribution $p \in \Delta([n])$ and parameter $\varepsilon \in (0,1]$, and granted sample access to an unknown distribution $q \in \Delta([n])$, one can test $p = q$ vs. $\|p - q\|_1 > \varepsilon$ with $O\big(\max\big(\frac{t_{c\varepsilon}}{\varepsilon^2}, \frac{1}{\varepsilon}\big)\big)$ samples from q. (Moreover, one can take $c = \frac{1}{18}$).*

7.2.1 High-level idea

As discussed in Section 2.4, the starting point of the proof is the connection between the K-functional and the "Q-norm" obtained in Lemma 18: indeed, this result ensures that for $T = 2t^2_{O(\varepsilon)}$, there exists a partition of the domain into sets $A_1, \dots, A_T$ such that

$$1 - O(\varepsilon) \le \|p\|_{Q(T)} = \sum_{j=1}^{T} \sqrt{\sum_{i \in A_j} p_i^2} = \sum_{j=1}^{T} \|p_{A_j}\|_2$$

where p_{A_j} is the restriction of the sequence p to the indices in A_j. But by the monotonicity of ℓ_p norms, we know that $\sum_{j=1}^{T} \|p_{A_j}\|_2 \le \sum_{j=1}^{T} \|p_{A_j}\|_1 = \sum_{j=1}^{T} \sum_{i \in A_j} p_i = \|p\|_1 = 1$. Therefore, what we obtain is in fact that

$$0 \le \sum_{j=1}^{T} (\underbrace{\|p_{A_j}\|_1 - \|p_{A_j}\|_2}_{\ge 0}) \le O(\varepsilon).$$

Now, if the right-hand side were *exactly* 0, then this would imply $\|p_{A_j}\|_1 = \|p_{A_j}\|_2$ for all j, and thus that p has (at most) one non-zero element in each A_j. Therefore, testing identity to p would boil down to testing identity on a distribution with support size T, which can be done with $O(\sqrt{T}/\varepsilon^2)$ samples.

This is not actually the case, of course: the right-hand-side is only small and not exactly zero. Yet, one can show that a robust version of the above holds, making this intuition precise: in Lemma 31, we show that on average, *most* of the probability mass of p is concentrated on a single point from each A_j. This sparsity implies that testing identity to p on this set of T points is indeed enough – leading to the theorem.

7.2.2 Proof of Theorem 30

Let $p \in \Delta([n])$ be a fixed, known distribution, assumed monotone non-increasing without loss of generality: $p_1 \ge p_2 \ge \cdots \ge p_n$. Given $\varepsilon \in (0, 1/2)$, we let t_ε be as above, namely such that

$$\kappa_p(t_\varepsilon) \ge 1 - 2\varepsilon.$$

From this, it follows by Lemma 18 that

$$\|p\|_{Q(T)} \ge 1 - 2\varepsilon, \tag{15}$$

where we set $T \stackrel{\text{def}}{=} 2t^2_\varepsilon$. Choose $A_1, \dots, A_T$ to be a partition of $[n]$ achieving the maximum (since we are in the finite, discrete case) defining $\|p\|_{Q(T)}$; and let $\tilde{p}$ be the subdistribution

on T elements defined as follows. For each $j \in [T]$, choose $i_j \stackrel{\text{def}}{=} \arg\max_{i \in A_j} p_i$, and set $\tilde{p}(j) \stackrel{\text{def}}{=} p(i_j)$.

▶ **Lemma 31** (Sparsity Lemma). *There exists an absolute constant $\kappa > 0$ such that $\tilde{p}([T]) = \sum_{j=1}^{T} p(i_j) \geq 1 - \kappa\varepsilon$. (Moreover, one can take $\kappa \stackrel{\text{def}}{=} \frac{2}{3-\sqrt{7}} \simeq 5.65$.)*

Proof. Fix any $j \in [T]$, and for convenience let $A \stackrel{\text{def}}{=} A_j$. Write a^* for the maximum element for p in A, so that $p(i_j) = \max_{a \in A} p(a) = p(a^*)$. We have by monotonicity $p(A) \geq \sqrt{\sum_{a \in A} p(a)^2}$, and moreover, letting $\alpha \stackrel{\text{def}}{=} p(A) - p(a^*) = p(A \setminus \{a^*\})$,

$$p(A) - \sqrt{\sum_{a \in A} p(a)^2} = p(a^*) + \alpha - \sqrt{p(a^*)^2 + \sum_{a \neq a^*} p(a)^2} \geq p(a^*) + \alpha - \sqrt{p(a^*)^2 + \alpha^2}.$$

We let $s > 1$ be a (non-integer) parameter to be chosen later. Suppose first that $\alpha \leq \frac{s}{s+1} p(A)$, or equivalenty $\alpha \leq s p(a^*)$. In that case, we have

$$\begin{aligned} p(A) - \sqrt{\sum_{a \in A} p(a)^2} &\geq p(a^*) + \alpha - p(a^*)\sqrt{1 + \left(\frac{\alpha}{p(a^*)}\right)^2} \\ &\geq p(a^*) + \alpha - p(a^*)\left(1 + \frac{\sqrt{s^2+1} - 1}{s} \frac{\alpha}{p(a^*)}\right) \\ &= \left(1 - \frac{\sqrt{s^2+1} - 1}{s}\right) \alpha \stackrel{\text{def}}{=} L_1(s)\alpha \end{aligned}$$

where we relied on the inequality $\sqrt{1+x^2} \leq 1 + \frac{\sqrt{s^2+1}-1}{s} x$ for $x \in [0, s]$. However, if $\alpha > s p(a^*)$, then we have

$$\begin{aligned} p(A) - \sqrt{\sum_{a \in A} p(a)^2} &= p(a^*) + \alpha - \sqrt{p(a^*)^2 + \sum_{a \neq a^*} p(a)^2} \geq \alpha - \sqrt{\sum_{a \neq a^*} p(a)^2} \\ &\geq \alpha - \sqrt{\lfloor s \rfloor \left(\frac{\alpha}{s}\right)^2 + 1 \cdot \left(\alpha - \frac{\lfloor s \rfloor}{s}\alpha\right)^2} \\ &= \left(1 - \sqrt{\frac{\lfloor s \rfloor}{s^2} + \left(1 - \frac{\lfloor s \rfloor}{s}\right)^2}\right) \alpha \stackrel{\text{def}}{=} L_2(s)\alpha. \end{aligned}$$

using the fact that $p(a^*)$ is the maximum probability value of any element, so that the total α has to be spread among at least $\lfloor s \rfloor + 1$ elements (recall that s will be chosen not to be an integer). Optimizing these two bounds leads to the choice of $s \stackrel{\text{def}}{=} \frac{4+\sqrt{7}}{3} \notin \mathbb{N}$, for which $L_1(s) = L_2(s) = 3 - \sqrt{7} \simeq 0.35$.

Putting it together, we obtain, summing over all $j \in [T]$, that

$$\begin{aligned} 1 - \|p\|_{Q(T)} &= \sum_{j=1}^{T} p(A_j) - \sum_{j=1}^{T} \sqrt{\sum_{i \in A_j} p(i)^2} = \sum_{j=1}^{T} \left(p(A_j) - \sqrt{\sum_{i \in A_j} p(i)^2}\right) \\ &\geq (3 - \sqrt{7}) \sum_{j=1}^{T} \left(p(A_j) - p(i_j)\right) \\ &= (3 - \sqrt{7})\left(1 - \tilde{p}([T])\right) \end{aligned}$$

which implies $\tilde{p}([T]) \geq \frac{1}{3-\sqrt{7}} \|p\|_{Q(T)} - \frac{1}{3-\sqrt{7}} + 1 \geq 1 - \frac{2}{3-\sqrt{7}}\varepsilon$ by Eq. (15). ◀

▶ **Lemma 32.** *Fix p, ε as above, let $S \stackrel{\text{def}}{=} \{i_1, \ldots, i_T\}$ be the corresponding set of T elements, and take κ as in Lemma 31. For any $q \in \Delta([n])$, if (i) $\sum_{j=1}^{T} q(i_j) \geq 1 - (\kappa + \frac{1}{3})\varepsilon$ and (ii) $\sum_{j=1}^{T} \left| \frac{\tilde{p}(j)}{p(S)} - \frac{\tilde{q}(j)}{q(S)} \right| \leq \frac{1}{3}\varepsilon$, then $\|p - q\|_1 \leq (3\kappa + 1)\varepsilon$.*

Proof. Unrolling the definition, and as $p(\bar{S}) \leq \kappa\varepsilon$ by Lemma 31,

$$\begin{aligned}
\|p - q\|_1 &= \sum_{i=1}^{n} |p(i) - q(i)| = \sum_{j=1}^{T} |p(i_j) - q(i_j)| + \sum_{i \notin S} |p(i) - q(i)| \\
&\leq \sum_{j=1}^{T} |p(i_j) - q(i_j)| + p(\bar{S}) + q(\bar{S}) \\
&\leq \sum_{j=1}^{T} |p(i_j) - q(i_j)| + \kappa\varepsilon + (\kappa + \tfrac{1}{3})\varepsilon = \sum_{j=1}^{T} \left| p(S)\frac{\tilde{p}(j)}{p(S)} - q(S)\frac{\tilde{q}(j)}{q(S)} \right| + (2\kappa + \tfrac{1}{3})\varepsilon \\
&\leq p(S) \sum_{j=1}^{T} \left| \frac{\tilde{p}(j)}{p(S)} - \frac{\tilde{q}(j)}{q(S)} \right| + \sum_{j=1}^{T} \frac{\tilde{q}(j)}{q(S)} |p(S) - q(S)| + (2\kappa + \tfrac{1}{3})\varepsilon \\
&= p(S) \cdot \sum_{j=1}^{T} \left| \frac{\tilde{p}(j)}{p(S)} - \frac{\tilde{q}(j)}{q(S)} \right| + |p(S) - q(S)| + (2\kappa + \tfrac{1}{3})\varepsilon \\
&\leq \tfrac{1}{3}\varepsilon + (\kappa + \tfrac{1}{3})\varepsilon + (2\kappa + \tfrac{1}{3})\varepsilon = (3\kappa + 1)\varepsilon
\end{aligned}$$

concluding the proof of the lemma. ◀

Let $\kappa > 0$ be the constant from Lemma 31. We let $\varepsilon' \stackrel{\text{def}}{=} \frac{\varepsilon}{3\kappa+1}$, and $T \stackrel{\text{def}}{=} 2t^2_{\varepsilon'}$, $\{i_1, \ldots, i_T\} \subseteq [n]$ the corresponding value and elements (i.e., T and the i_j's are as in the foregoing discussion (chosen with regard to ε' and the known distribution p)). For convenience, denote by $\tilde{q}$ the (unknown) subdistribution on $[T]$ defined by $\tilde{q}(j) \stackrel{\text{def}}{=} q(i_j)$ for $j \in [T]$.

We first verify that $\tilde{q}([T]) \geq 1 - \kappa\varepsilon'$, with $O(1/\varepsilon')$ samples (specifically, we distinguish, with probability at least $9/10$, between $\tilde{q}([T]) \geq 1 - \kappa\varepsilon'$ and $\tilde{q}([T]) \leq 1 - (\kappa + \frac{1}{3})\varepsilon'$; and reject in the latter case). Once this is done, we apply one of the known identity testing algorithms to $\bar{p}, \bar{q} \in \Delta([T])$, renormalized versions of $\tilde{p}, \tilde{q}$:

$$\bar{p} = \frac{\tilde{p}}{\tilde{p}([T])}, \quad \bar{q} = \frac{\tilde{q}}{\tilde{q}([T])}$$

using rejection sampling (note that we have the explicit description of $\bar{p}$; and, since $\tilde{q}([T]) \geq 1 - (\kappa + \frac{1}{3})\varepsilon'$ (conditioning on the first test meeting its guarantee), we can obtain m independent samples from $\bar{q}$ with an expected $O(m)$ number of samples from q). This is done with parameter ε' and failure probability $1/10$; and costs $O\left(\frac{\sqrt{T}}{\varepsilon'^2}\right) = O\left(\frac{t_{\varepsilon'}}{\varepsilon'^2}\right)$ samples from q.

Turning to the correctness: we condition on both tests meeting their guarantees, which by a union bound holds with probability at least $4/5$.

- If $p = q$, then $q(S) = p(S) \geq 1 - \kappa\varepsilon'$, and $\bar{q} = \bar{p}$: neither the first nor the second test reject, and the overall algorithm accepts.
- If the algorithm accepts, then $q(S) \geq 1 - (\kappa + \frac{1}{3})\varepsilon'$ (by the first test) and $\sum_{j=1}^{T} \left| \frac{\tilde{p}(j)}{p(S)} - \frac{\tilde{q}(j)}{p(S)} \right| \leq \varepsilon'$ (by the second): Lemma 32 then guarantees that $\|p - q\|_1 \leq 3\kappa + 1\varepsilon' = \varepsilon$.

Observing that for $\kappa = \frac{2}{3-\sqrt{7}}$ (as suggested by Lemma 31) we have $3\kappa + 1 \leq 18$ establishes the last part of the theorem.

7.3 Tightening the Lower Bound

As a last step, one may want to strenghten the lower bound obtained by the communication complexity reduction of Theorem 24. We here describe how this can be achieved using more standard arguments from distribution testing. However, we stress that these arguments in some sense are applicable "'after the fact," that is after Section 7.1 revealed the connection to the K-functional, and the bound we should aim for. Specifically, we prove the following:

▶ **Theorem 33.** *For any $p \in \Delta([n])$, and any $\varepsilon \in (0, 1/2)$ any algorithm testing identity to p must have sample complexity $\Omega\left(\frac{t_\varepsilon}{\varepsilon}\right)$.*

Proof. Fix $p \in \Delta([n])$ and $\varepsilon \in (0, 1/2)$ as above, and consider the corresponding value t_ε; we assume that $t_\varepsilon \geq 2$, as otherwise there is nothing to prove.[13] Without loss of generality – as we could always consider a sufficiently small approximation, and take the limit in the end, we further assume the infimum defining κ_p is attained: let $h, \ell \in [0,1]^n$ be such that $p = h + \ell$ and $\kappa_p(t_\varepsilon) = \|h\|_1 + t_\varepsilon \|\ell\|_2 = 1 - 2\varepsilon$.

Since $\|\ell\|_1 = 1 - \|h\|_1$, from the definition of h, ℓ, we have that $1 - 2\varepsilon = 1 - \|\ell\|_1 + t_\varepsilon \|\ell\|_2$, from which

$$0 < \|\ell\|_2 = \frac{\|\ell\|_1 - 2\varepsilon}{t_\varepsilon} \leq \frac{1}{t_\varepsilon} \tag{16}$$

(note that the right inequality is strict because $\varepsilon > 0$: since if $\|\ell\|_2 = 0$, then $\|\ell\|_1 = 0$ and $h = p$; but then $\kappa_{t_\varepsilon} = \|p\|_1 = 1$.) In particular, this implies $\|\ell\|_1 - 2\varepsilon > 0$.

With this in hand, we will apply the following theorem, due to Valiant and Valiant:

▶ **Theorem 34** ([53, Theorem 4])**.** *Given a distribution $p \in \Delta([n])$, and associated values $(\varepsilon_i)_{i \in [n]}$ such that $\varepsilon_i \in [0, p_i]$ for each i, define the distribution over distributions $\mathcal{Q}$ by the process: independently for each domain element i, set uniformly at random $q_i = p_i \pm \varepsilon_i$, and then normalize q to be a distribution. Then there exists a constant $c > 0$ such that is takes at least $c\left(\sum_{i=1}^n \varepsilon_i^4 / p_i^2\right)^{-1/2}$ samples to distinguish p from $\mathcal{Q}$ with success probability $2/3$. Further, with probability at least $1/2$ the ℓ_1 distance between p and a uniformly random distribution from $\mathcal{Q}$ is at least $\min\left(\sum_{i=1}^n \varepsilon_i - \max_i \varepsilon_i, \frac{1}{2}\sum_{i=1}^n \varepsilon_i\right)$.*

We want to invoke the above theorem with ℓ being, roughly speaking, the "random perturbation" to p. Indeed, since ℓ has small ℓ_2 norm of order $O(1/t_\varepsilon)$ by (16) (which gives a good lower bound) and has ℓ_1 sum $\Omega(\varepsilon)$ (which gives distance), this seems to be a natural choice.

In view of this, set $\alpha \stackrel{\text{def}}{=} \frac{2\varepsilon}{\|\ell\|_1} \in (0,1)$ and, for $i \in [n]$, $\varepsilon_i \stackrel{\text{def}}{=} \alpha \ell_i \leq \ell_i \in [0, p_i]$. Theorem 33 will then be a direct consequence of the next two claims:

▶ **Claim 35** (Distance)**.** *We have $\min\left(\sum_{i=1}^n \varepsilon_i - \max_i \varepsilon_i, \frac{1}{2}\sum_{i=1}^n \varepsilon_i\right) \geq \varepsilon$.*

Proof. Since by our choice of α it is immediate that $\sum_{i=1}^n \varepsilon_i = \frac{2\varepsilon}{\|\ell\|_1} \sum_{i=1}^n \ell_i = 2\varepsilon$, it suffices to show that $\max_i \varepsilon_i \leq \varepsilon$, or equivalently that $\max_i \ell_i \leq \frac{1}{2}\|\ell\|_1$. But this follows from the fact that $\|\ell\|_\infty \leq \|\ell\|_2 \leq \frac{\|\ell\|_1}{t_\varepsilon}$, and our assumption that $t_\varepsilon \geq 2$. ◀

It then remains to analyze the lower bound obtained through the application of Theorem 34:

▶ **Claim 36** (Lower bound)**.** *With the ε_i's defined as before, $\left(\sum_{i=1}^n \varepsilon_i^4 / p_i^2\right)^{-1/2} \geq \frac{2t_\varepsilon}{\varepsilon}$.*

[13] Indeed, an immediate lower bound of $\Omega(1/\varepsilon)$ on this problem holds.

Proof. Unrolling the definition of the ε_i's,

$$\sum_{i=1}^{n} \frac{\varepsilon_i^4}{p_i^2} = \alpha^4 \sum_{i=1}^{n} \frac{\ell_i^4}{p_i^2} = \alpha^4 \sum_{i=1}^{n} \frac{\ell_i^2}{p_i^2} \ell_i^2 \leq \alpha^4 \sum_{i=1}^{n} \ell_i^2 = \frac{2^4 \varepsilon^4}{\|\ell\|_1^4} \|\ell\|_2^2 = \left(\frac{4\varepsilon^2}{\|\ell\|_1^2} \frac{\|\ell\|_1 - 2\varepsilon}{t_\varepsilon} \right)^2$$

where the last equality is (16). This yields

$$\left(\sum_{i=1}^{n} \frac{\varepsilon_i^4}{p_i^2} \right)^{-1/2} \geq \frac{t_\varepsilon}{4\varepsilon^2} \cdot \frac{\|\ell\|_1^2}{\|\ell\|_1 - 2\varepsilon} = \frac{t_\varepsilon}{2\varepsilon} \cdot \frac{\left(\frac{\|\ell\|_1}{2\varepsilon}\right)^2}{\frac{\|\ell\|_1}{2\varepsilon} - 1} \geq \frac{2t_\varepsilon}{\varepsilon}$$

where the last inequality comes from $f : x > 1 \mapsto \frac{x^2}{x-1}$ achieving its minimum, 4, at $x = 2$. ◀

Combining the two claims with Theorem 34 implies, by a standard argument, the lower bound of Theorem 33. ◀

▶ Remark. A straightforward modification of the proof of Theorem 33 allows one to prove a somewhat more general statement, namely a lower bound of $\Omega\left(\gamma t_\gamma / \varepsilon^2\right)$ for any $\gamma \in [\varepsilon, 1/2]$ such that $t_\gamma \geq 2$. In particular, this implies an incomparable bound of $\Omega\left(t_{1/4}/\varepsilon^2\right)$ as long as p does not put almost all its probability weight on $O(1)$ elements.

On the optimality of our bound. We conclude this section by briefly discussing the optimality of our bound, and specifically whether one could hope to strenghten Theorem 33 to obtain an $\Omega\left(t_\varepsilon/\varepsilon^2\right)$ lower bound. Unfortunately, it is easy to come up with simple (albeit contrived) counterexamples: e.g., fix $\varepsilon \in (0, 1/3)$, and let $p \in \Delta([n])$ be the distribution that puts mass $1 - 3\varepsilon$ on the first element and uniformly spreads the rest among the remaining $n - 1$ elements. A straightforward calculation shows that, for this distribution $p = p(\varepsilon)$, one has $\kappa_p^{-1}(1 - 2\varepsilon) = \Theta(\sqrt{n})$; and it is not hard to check that one can indeed test identity to p with $O(\sqrt{n}/\varepsilon)$ samples only,[14] and so the $\Omega(t_\varepsilon/\varepsilon)$ lower bound is tight in this case.

Although this specific instance is somewhat unnatural, as it fails to be a counterexample for any distance parameter $\varepsilon' \ll \varepsilon$, it does rule out an improvement of Theorem 33 for the full range of parameters. On the other hand, it is also immediate to see that the upper bound $O\left(t_\varepsilon/\varepsilon^2\right)$ cannot be improved in general, as demonstrated by choosing p to be the uniform distribution (yet, in this case, the extension provided by the above remark does provide the optimal bound).

8 Lower Bounds on Other Properties

In this section we demonstrate how our methodology can be used to easily obtain lower bounds on the sample complexity of various properties of distributions. To this end, we provide sketches of proofs of lower bounds for monotonicity testing, k-modality, and the "symmetric sparse support" property (that we define below). We remark that using minor variations on the reductions presented in Section 5 and Section 7, it is also straightforward to obtain lower bounds for properties of distributions such as being binomially distributed, Poisson binomially distributed, and having a log-concave probability mass function. Throughout this section, we fix ε to be a small constant and refer to testing with respect to proximity $\Theta(\varepsilon)$.

[14] Indeed, any distribution q such that $\|q - p\|_1 > \varepsilon$ must either be such that $|p(1) - q(1)| = \Omega(\varepsilon)$ or $\left|p|_{[n]\setminus\{1\}} - q|_{[n]\setminus\{1\}}\right| = \Omega(1)$. The first case only takes $O(1/\varepsilon)$ samples, while the second can be achieved by rejection sampling with $O(1/\varepsilon) \cdot O(\sqrt{n})$ samples.

Monotonicity on the integer line and the Boolean hypercube. We start with the problem of testing monotonicity on the integer line, that is, testing whether a distribution $p \in \Delta([n])$ has a monotone probability mass function. Consider the "vanilla" reduction, presented in Section 5. Note that for yes-instances, we obtain the uniform distribution, which is monotone. For no-instances, however, we obtain a distribution p that has mass $1/n$ on a $(1-\varepsilon)$-fraction of the domain, is unsupported on a $(\varepsilon/2)$-fraction of the domain, and has mass $2/n$ on the remaining $(\varepsilon/2)$-fraction. Typically, p is $\Omega(1)$-far from being monotone; however, it could be the case that the first (respectively, last) $\varepsilon n/2$ elements are of 0 mass, and the last (respectively, first) $\varepsilon n/2$ elements are of mass $2/n$, in which case p is perfectly monotone. To remedy this, all we have to do is let the referee emulate a distribution $p' \in \Delta([3n])$ such that $p'_i = \begin{cases} \frac{1}{3}p_{i-n} & i \in \{n+1,\ldots,2n\} \\ \frac{1}{3n} & \text{otherwise} \end{cases}$. It is immediate to see that the probability mass functions of p' is $(\varepsilon/3)$-far from monotone.

The idea above can be extended to monotonicity over the hypercube as follows. We start with the uniformity reduction, this time over the domain $\{0,1\}^n$. As before, yes-instances will be mapped to the uniform distribution over the hypercube, which is monotone, and no-instances will be mapped to a distribution that has mass $1/2^n$ on a $(1-\varepsilon)$-fraction of the domain, is unsupported on a $(\varepsilon/2)$-fraction of the domain, and has mass $1/2^{n-1}$ on the remaining $(\varepsilon/2)$-fraction – but could potentially be monotonously *strictly* increasing (or decreasing). This time, however, the "boundary" is larger than the "edges" of the integer line, and we cannot afford to pad it with elements of weight $1/2^n$. Instead, the referee, who receives for the players samples drawn from a distribution $p \in \Delta(\{0,1\}^n)$, emulates a distribution $p'' \in \Delta\big(\{0,1\}^{n+1}\big)$ over a larger hypercube whose additional coordinate determines between a negated or regular copy of p; that is, $p'(z) = \begin{cases} p(z) & z_1 = 0 \\ \frac{1}{2^{n-1}} - p(z) & z_1 = 1 \end{cases}$ (where the referee chooses $z_1 \in \{0,1\}$ independently and uniformly at random for each new sample). Hence, even if p is monotonously increasing (or decreasing), the emulated distribution p'' is $\Omega(\varepsilon)$-far from monotone. By the above, we obtain $\tilde{\Omega}(\sqrt{n})$ and $\tilde{\Omega}(2^{n/2})$ lower bounds on the sample complexity of testing monotonicity on the line and on the hypercube, respectively.

***k*-modality.** Recall that a distribution $p \in \Delta([n])$ is said to be *k-modal* if its probability mass function has at most k "peaks" and "valleys." Such distributions are natural generalizations of monotone (for $k = 0$) and unimodal (for $k = 1$) distributions. Fix a sublinear k, and consider the uniformity reduction presented in Section 5, with the additional step of letting the prover apply a random permutation to the domain $[n]$ (similarly to the reduction shown in Section 5.1). Note that yes-instances are still mapped to the uniform distribution (which is clearly k-modal), and no-instances are mapped to distributions with mass $1/n$, $2/n$, and 0 on a $(1-\varepsilon)$, $(\varepsilon/2)$, and $(\varepsilon/2)$ (respectively) fractions of the domain. Intuitively, applying a random permutation of the domain to such a distribution "spreads" the elements with masses 0 and $2/n$ nearly uniformly, causing many level changes (i.e., high modality); indeed, it is straightforward to verify that with high probability over the choice of a random permutation of the domain, such a distribution will indeed be $\Omega(\varepsilon)$-far from k-modal. This yields an $\tilde{\Omega}(\sqrt{n})$ lower bound on the sample complexity of testing k-modality, nearly matching the best known lower bound of $\Omega(\max(\sqrt{n}, k/\log k))$ following from [19], for $k/\log(k) = O(\sqrt{n})$.

Symmetric sparse support. Consider the property of distributions $p \in \Delta([n])$ such that when projected to its support, p is mirrored around the middle of the domain. That is, p is said to have a *symmetric sparse support* if there exists $S = \{i_0 < i_2 < \cdots < i_{2\ell}\} \subseteq [n]$ with

$i_\ell = \frac{n}{2}$ such that: (1) $p(i) = 0$ for all $i \in [n] \setminus S$, and (2) $p(i_{\ell+1-j}) = p(i_{\ell+j})$ for all $0 \leq j \leq \ell$. We sketch a proof of an $\tilde{\Omega}(\sqrt{n})$ lower bound on the sample complexity of testing this property. Once again, we shall begin with the uniformity reduction presented in Section 5, obtaining samples from a distribution $p \in \Delta([n/2])$. Then the referee emulates samples from the distribution $p' \in \Delta([n])$ that is distributed as p on its left half, and uniformly distributed on its right half; that is, $p'_i = \begin{cases} p_i/2 & i \in [n] \\ 1/n & \text{otherwise} \end{cases}$. Note that yes-instances are mapped to the uniform distribution, which has symmetric sparse support, and no-instances are mapped to distributions in which the right half is uniformly distributed and the left half contains $\varepsilon n/2$ elements of mass $2/n$, and hence it is $\Omega(\varepsilon)$-far from having symmetric sparse support.

Other properties. As aforementioned, similar techniques as in the reductions above (as well as in the identity testing reduction of Section 7, invoked on a specific p, e.g., the $\mathrm{Bin}(n, 1/2)$ distribution) can be applied to obtain nearly-tight lower bounds of $\tilde{\Omega}(\sqrt{n})$ (respectively $\tilde{\Omega}(n^{1/4})$) for the properties of being log-concave and monotone hazard rate (respectively Binomially and Poisson Binomially distributed). See e.g., [20] for the formal definitions of these properties.

9 Testing with Conditional Samples

In this section we show that reductions from communication complexity protocols can be used to obtain lower bounds on the sample complexity of distribution testers that are augmented with conditional samples. These testing algorithms, first introduced in [22, 21], aim to address scenarios that arise both in theory and practice yet are not fully captured by the standard distribution testing model.

In more detail, algorithms for testing with conditional samples are distribution testers that, in addition to sample access to a distribution $p \in \Delta(\Omega)$, can ask for samples from p conditioned on the sample belonging to a subset $S \subseteq \Omega$. It turns out that testers with conditional samples are much stronger than standard distribution testers, leading in many cases to exponential savings (or even more) in the sample complexity. In fact, these testing algorithms can often maintain their power even if they only have the ability to query subsets of a particular structure.

One of the most commonly studied restricted conditional samples models is the PAIRCOND model [21]. In this model, the testers can either obtain standard samples from p, or specify two distinct indices $i, j \in \Omega$ and get a sample from p conditioned on membership in $S = \{i, j\}$. As shown in [21, 18], even under this restriction one can obtain constant- or $\mathrm{poly}\log(n)$-query testers for many properties, such as uniformity, identity, closeness, and monotonicity (all of which require $\Omega(\sqrt{n})$ or more samples in the standard sampling setting). This, along with the inherent difficulty of proving hardness results against *adaptive* algorithms, makes proving lower bounds in this setting a challenging task; and indeed the PAIRCOND lower bounds established in the aforementioned works are quite complex and intricate.

We will prove, via a reduction from communication complexity, a strong lower bound on the sample complexity of any PAIRCOND algorithm for testing *junta distributions*, a class of distributions introduced in [5] (see definition below).

Since PAIRCOND algorithms are stronger than standard distribution testers (in particular, they can make adaptive queries), we shall reduce from the general randomized communication complexity model (rather than from the SMP model, as we did for standard distribution testers). In this model, Alice and Bob are given inputs x and y as well as a common random

string, and the parties aim to compute a function $f(x, y)$ using the minimum amount of communication.

We say that a distribution $p \in \Delta(\{0,1\}^n)$ is a *k-junta distribution* (with respect to the uniform distribution) if its probability mass function is only influenced by k of its variables. We outline below a proof of the following lower bound.

▶ **Theorem 37.** *Every* PAIRCOND *algorithm for testing k-junta distributions must make $\Omega(k)$ queries.*

Sketch of Proof. We closely follow the k-linearity lower bound in [15] and reduce from the unique $(k/2)$-disjointness problem. In this promise problem, Alice and Bob get inputs $x \in \{0,1\}^n$ and $y \in \{0,1\}^n$ (respectively) of Hamming weight $k/2$ each, and the parties are required to decide whether $\sum_{i=1}^n x_i y_i = 1$ or $\sum_{i=1}^n x_i y_i = 0$. It is well-known that in every randomized protocol for this problem the parties must communicate $\Omega(k)$ bits.

Assume there exists a PAIRCOND algorithm for testing k-junta distributions, with query complexity q. The reduction is as follows. Alice sets $A = \{\, i \in [n] : x_i = 1 \,\}$ and considers the character function $\chi_A(z) = \oplus_{i \in A} z_i$, and similarly Bob sets $B = \{\, i \in [n] : y_i = 1 \,\}$ and considers the character function $\chi_B(z) = \oplus_{i \in B} z_i$. Both players then invoke the tester for k-junta distributions, feeding it samples emulated from the distribution $p \in \Delta(\{0,1\}^n)$ given by $p(z) = \chi_{A \triangle B}(z)/2^{n-1}$ (where $\chi_{A \triangle B}(z) = \oplus_{i \in A \triangle B} z_i$); note that since the non-zero character functions are balanced, p is indeed a probability distribution. Recall that each query of a PAIRCOND algorithm is performed by either setting $S = \{0,1\}^n$, or choosing $z, z' \in \{0,1\}^n$ and setting $S = \{z, z'\}$, then sampling from $p|_S$. The players emulate each PAIRCOND query by the following rejection sampling procedure:

Sampling query ($S = \{0,1\}^n$): Alice and Bob proceed as follows.

1. Choose $z \in S$ uniformly at random, using shared randomness;
2. Exchange $\chi_A(z)$ and $\chi_B(z)$ between the players, and compute $\chi_{A \triangle B}(z) = \chi_A(z) \cdot \chi_B(z)$;
3. If $\chi_{A \triangle B}(z) = 1$, feed the tester with the sample z. Otherwise repeat the process.

Note that since $\chi_{A \triangle B}(z)$ is a balanced function, then on expectation each PAIRCOND query to p can be emulated by exchanging $O(1)$ bits.

Pairwise query ($S = \{z, z'\}$ for some $z, z' \in \{0,1\}^n$): exchange $\chi_A(z), \chi_A(z')$ and $\chi_B(z), \chi_B(z')$ between the players, compute $\chi_{A \triangle B}(z)$ and $\chi_{A \triangle B}(z')$, and use shared randomness to sample from S with the corresponding (now fully known) conditional probabilities.

The above gives a protocol with *expected* communication complexity $O(q)$, correct with probability $5/6$. To convert it to a honest-to-goodness protocol with communication complexity $O(q)$ and success probability $2/3$, it suffices for Alice and Bob to run the above protocol and stop (and output reject) as soon as they go over Ck bits of communication, for some absolute constant $C > 0$. An application of Markov's inequality guarantees that this happens with probability at most $1/6$, yielding the claimed bound on the error probability of the protocol.

Finally, note that on the one hand, if (x, y) is such that $\sum_{i=1}^n x_i y_i = 0$, then $\chi_{A \triangle B}(z)$ is a degree-k character, and in particular, a k-junta. Hence, by definition p is a k-junta distribution. On the other hand, if (x, y) is such that $\sum_{i=1}^n x_i y_i = 1$, then $\chi_{A \triangle B}(z)$ is a degree-$(k-2)$ character, which in particular disagrees with every k-junta on $\Omega(1)$-fraction of the inputs. Therefore, since p is uniform over its support, we can deduce that that p is $\Omega(1)$-far in ℓ_1-distance from any k-junta distribution. ◀

Acknowledgments. We thank Oded Goldreich and Rocco Servedio for insightful conversations and for technical and conceptual suggestions regarding the contents of this work and its presentation.

References

1. Jayadev Acharya, Clément L. Canonne, and Gautam Kamath. A Chasm Between Identity and Equivalence Testing with Conditional Queries. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2015)*, volume 40 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 449–466. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2015. doi:10.4230/LIPIcs.APPROX-RANDOM.2015.449.
2. Jayadev Acharya, Hirakendu Das, Ashkan Jafarpour, Alon Orlitsky, and Shengjun Pan. Competitive closeness testing. In *Proceedings of COLT*, pages 47–68, 2011.
3. Jayadev Acharya and Constantinos Daskalakis. Testing Poisson Binomial Distributions. In *Proceedings of SODA*, pages 1829–1840, 2015.
4. Jayadev Acharya, Constantinos Daskalakis, and Gautam Kamath. Optimal testing for properties of distributions. In *Proceedings of NIPS*, pages 3577–3598, 2015.
5. Maryam Aliakbarpour, Eric Blais, and Ronitt Rubinfeld. Learning and testing junta distributions. In *Proceedings of COLT*, volume 49 of *JMLR Workshop and Conference Proceedings*, pages 19–46. JMLR.org, 2016.
6. Sergey V. Astashkin. Rademacher functions in symmetric spaces. *Journal of Mathematical Sciences*, 169(6):725–886, sep 2010. doi:10.1007/s10958-010-0074-z.
7. Tuğkan Batu, Sanjoy Dasgupta, Ravi Kumar, and Ronitt Rubinfeld. The complexity of approximating the entropy. *SIAM Journal on Computing*, 35(1):132–150, 2005.
8. Tuğkan Batu, Eldar Fischer, Lance Fortnow, Ravi Kumar, Ronitt Rubinfeld, and Patrick White. Testing random variables for independence and identity. In *Proceedings of FOCS*, pages 442–451, 2001.
9. Tuğkan Batu, Lance Fortnow, Ronitt Rubinfeld, Warren D. Smith, and Patrick White. Testing that distributions are close. In *Proceedings of FOCS*, pages 189–197, 2000.
10. Tuğkan Batu, Lance Fortnow, Ronitt Rubinfeld, Warren D. Smith, and Patrick White. Testing closeness of discrete distributions. *ArXiV*, abs/1009.5397, 2010. This is a long version of [9].
11. Tuğkan Batu, Ravi Kumar, and Ronitt Rubinfeld. Sublinear algorithms for testing monotone and unimodal distributions. In *Proceedings of STOC*, pages 381–390, New York, NY, USA, 2004. ACM. doi:10.1145/1007352.1007414.
12. Colin Bennett and Robert C. Sharpley. *Interpolation of Operators*. Pure and Applied Mathematics. Elsevier Science, 1988. URL: https://books.google.com/books?id=HpqF9zjZWMMC.
13. Bhaswar B. Bhattacharya and Gregory Valiant. Testing closeness with unequal sized samples. In *Proceedings of NIPS*, pages 2611–2619, 2015.
14. Arnab Bhattacharyya, Eldar Fischer, Ronitt Rubinfeld, and Paul Valiant. Testing monotonicity of distributions over general partial orders. In *Proceedings of ITCS*, pages 239–252, 2011. URL: http://conference.itcs.tsinghua.edu.cn/ICS2011/content/papers/38.html.
15. Eric Blais, Joshua Brody, and Kevin Matulef. Property testing lower bounds via communication complexity. *Computational Complexity*, 21(2):311–358, 2012. doi:10.1007/s00037-012-0040-x.
16. Joshua Brody, Kevin Matulef, and Chenggang Wu. Lower bounds for testing computability by small width OBDDs. In *TAMC*, volume 6648 of *Lecture Notes in Computer Science*, pages 320–331. Springer, 2011.

17 Clément L. Canonne. Big Data on the Rise? Testing Monotonicity of Distributions. In *Proceedings of ICALP*, pages 294–305. Springer, 2015. doi:10.1007/978-3-662-47672-7_24.

18 Clément L. Canonne. A Survey on Distribution Testing: your data is Big. But is it Blue? *Electronic Colloquium on Computational Complexity (ECCC)*, 22:63, April 2015. URL: https://eccc.weizmann.ac.il/report/2015/063/.

19 Clément L. Canonne. Are Few Bins Enough: Testing Histogram Distributions. In *Proceedings of PODS*. Association for Computing Machinery (ACM), 2016. doi:10.1145/2902251.2902274.

20 Clément L. Canonne, Ilias Diakonikolas, Themis Gouleakis, and Ronitt Rubinfeld. Testing Shape Restrictions of Discrete Distributions. In *33rd Symposium on Theoretical Aspects of Computer Science (STACS 2016)*, volume 47 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 25:1–25:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.STACS.2016.25.

21 Clément L. Canonne, Dana Ron, and Rocco A. Servedio. Testing probability distributions using conditional samples. *SIAM Journal on Computing*, 44(3):540–616, 2015. Also available on arXiv at abs/1211.2664. doi:10.1137/130945508.

22 Sourav Chakraborty, Eldar Fischer, Yonatan Goldhirsh, and Arie Matsliah. On the power of conditional samples in distribution testing. In *Proceedings of ITCS*, pages 561–580, New York, NY, USA, 2013. ACM. doi:10.1145/2422436.2422497.

23 Siu-On Chan, Ilias Diakonikolas, Gregory Valiant, and Paul Valiant. Optimal algorithms for testing closeness of discrete distributions. In *Proceedings of SODA*, pages 1193–1203. Society for Industrial and Applied Mathematics (SIAM), 2014.

24 Constantinos Daskalakis, Ilias Diakonikolas, Rocco A. Servedio, Gregory Valiant, and Paul Valiant. Testing k-modal distributions: Optimal algorithms via reductions. In *Proceedings of SODA*, pages 1833–1852. Society for Industrial and Applied Mathematics (SIAM), 2013. URL: http://dl.acm.org/citation.cfm?id=2627817.2627948.

25 Ilias Diakonikolas and Daniel M. Kane. A new approach for testing properties of discrete distributions. In *Proceedings of FOCS*. IEEE Computer Society, 2016.

26 Ilias Diakonikolas, Daniel M. Kane, and Vladimir Nikishkin. Optimal Algorithms and Lower Bounds for Testing Closeness of Structured Distributions. In *Proceedings of FOCS*. Institute of Electrical and Electronics Engineers (IEEE), oct 2015. doi:10.1109/focs.2015.76.

27 Ilias Diakonikolas, Daniel M. Kane, and Vladimir Nikishkin. Testing Identity of Structured Distributions. In *Proceedings of SODA*, pages 1841–1854. Society for Industrial and Applied Mathematics (SIAM), January 2015.

28 Moein Falahatgar, Ashkan Jafarpour, Alon Orlitsky, Venkatadheeraj Pichapathi, and Ananda Theertha Suresh. Faster algorithms for testing under conditional sampling. *ArXiV*, abs/1504.04103, April 2015.

29 Eldar Fischer, Oded Lachish, and Yadu Vasudev. Improving and extending the testing of distributions for shape-restricted properties. *ArXiV*, abs/1609.06736, September 2016.

30 Oded Goldreich. The uniform distribution is complete with respect to testing identity to a fixed distribution. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:15, 2016.

31 Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, 45(4):653–750, July 1998.

32 Oded Goldreich and Dana Ron. On testing expansion in bounded-degree graphs. *Electronic Colloquium on Computational Complexity (ECCC)*, 7:20, 2000. URL: https://eccc.weizmann.ac.il/report/2000/020/.

33 Paweł Hitczenko and Stanisław Kwapień. On the Rademacher series. In *Probability in Banach spaces, 9 (Sandjberg, 1993)*, volume 35 of *Progr. Probab.*, pages 31–36. Birkhäuser

Boston, Boston, MA, 1994.

34 Tord Holmstedt. Interpolation of Quasi-Normed Spaces. *Mathematica Scandinavica*, 26(0):177–199, 1970. URL: `http://www.mscand.dk/article/view/10976`.

35 Piotr Indyk, Reut Levi, and Ronitt Rubinfeld. Approximating and Testing k-Histogram Distributions in Sub-linear Time. In *Proceedings of PODS*, pages 15–22, 2012.

36 Jiantao Jiao, Kartik Venkat, and Tsachy Weissman. Order-optimal estimation of functionals of discrete distributions. *ArXiV*, abs/1406.6956, 2014. URL: `http://arxiv.org/abs/1406.6956`.

37 Reut Levi, Dana Ron, and Ronitt Rubinfeld. Testing properties of collections of distributions. *Theory of Computing*, 9:295–347, 2013. `doi:10.4086/toc.2013.v009a008`.

38 Stephen J. Montgomery-Smith. The distribution of Rademacher sums. *Proceedings of the American Mathematical Society*, 109(2):517–522, 1990.

39 Ilan Newman. Property testing of massively parametrized problems – A survey. In *Property Testing*, volume 6390 of *Lecture Notes in Computer Science*, pages 142–157. Springer, 2010.

40 Ilan Newman and Mario Szegedy. Public vs. private coin flips in one round communication games. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 561–570. ACM, 1996.

41 Liam Paninski. Estimating entropy on m bins given fewer than m samples. *IEEE Transactions on Information Theory*, 50(9):2200–2203, 2004.

42 Liam Paninski. A coincidence-based test for uniformity given very sparsely sampled discrete data. *IEEE Transactions on Information Theory*, 54(10):4750–4755, 2008. `doi:10.1109/TIT.2008.928987`.

43 Jaak Peetre. *A theory of interpolation of normed spaces*. Notas de Matemática, No. 39. Instituto de Matemática Pura e Aplicada, Conselho Nacional de Pesquisas, Rio de Janeiro, 1968.

44 David Pollard. Asymptopia. `http://www.stat.yale.edu/~pollard/Books/Asymptopia`, 2003. Manuscript.

45 Sofya Raskhodnikova, Dana Ron, Amir Shpilka, and Adam Smith. Strong lower bounds for approximating distributions support size and the distinct elements problem. *SIAM Journal on Computing*, 39(3):813–842, 2009.

46 Ronitt Rubinfeld. Taming big probability distributions. *XRDS: Crossroads, The ACM Magazine for Students*, 19(1):24, sep 2012. `doi:10.1145/2331042.2331052`.

47 Ronitt Rubinfeld and Rocco A. Servedio. Testing monotone high-dimensional distributions. *Random Structures and Algorithms*, 34(1):24–44, January 2009. `doi:10.1002/rsa.v34:1`.

48 Ronitt Rubinfeld and Madhu Sudan. Robust characterization of polynomials with applications to program testing. *SIAM Journal on Computing*, 25(2):252–271, 1996.

49 Gregory Valiant and Paul Valiant. A CLT and tight lower bounds for estimating entropy. *Electronic Colloquium on Computational Complexity (ECCC)*, 17:179, 2010. URL: `https://eccc.weizmann.ac.il/report/2010/179/`.

50 Gregory Valiant and Paul Valiant. Estimating the unseen: A sublinear-sample canonical estimator of distributions. *Electronic Colloquium on Computational Complexity (ECCC)*, 17:180, 2010. URL: `https://eccc.weizmann.ac.il/report/2010/180/`.

51 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 STOC*, pages 685–694. ACM, 2011.

52 Gregory Valiant and Paul Valiant. The power of linear estimators. In *Proceedings of FOCS*, pages 403–412, October 2011. See also [49] and [50]. `doi:10.1109/FOCS.2011.81`.

53 Gregory Valiant and Paul Valiant. An automatic inequality prover and instance optimal identity testing. In *Proceedings of FOCS*, pages 51–60, 2014.

54 Paul Valiant. Testing symmetric properties of distributions. *SIAM Journal on Computing*,

40(6):1927–1968, 2011.

55 Wikipedia. Hypergeometric distribution – wikipedia, the free encyclopedia, 2016. [Online; accessed 28-May-2016]. URL: https://en.wikipedia.org/w/index.php?title=Hypergeometric_distribution&oldid=702419153#Multivariate_hypergeometric_distribution.

56 Yihong Wu and Pengkun Yang. Minimax rates of entropy estimation on large alphabets via best polynomial approximation. *ArXiV*, abs/1407.0381, 2014. URL: http://arxiv.org/abs/1407.0381.

57 Bin Yu. Assouad, Fano, and Le Cam. In *Festschrift for Lucien Le Cam*, pages 423–435. Springer, 1997. doi:10.1007/978-1-4612-1880-7_29.

Exponentially Small Soundness for the Direct Product Z-Test*

Irit Dinur[1] and Inbal Livni Navon[2]

1 **Weizmann Institute of Science, Rehovot, Israel**
irit.dinur@weizmann.ac.il
2 **Weizmann Institute of Science, Rehovot, Israel**
inbal.livni@weizmann.ac.il

Abstract

Given a function $f : [N]^k \to [M]^k$, the Z-test is a three query test for checking if a function f is a direct product, namely if there are functions $g_1, \ldots g_k : [N] \to [M]$ such that $f(x_1, \ldots, x_k) = (g_1(x_1), \ldots g_k(x_k))$ for every input $x \in [N]^k$.

This test was introduced by Impagliazzo et. al. (SICOMP 2012), who showed that if the test passes with probability $\epsilon > \exp(-\sqrt{k})$ then f is $\Omega(\epsilon)$ close to a direct product function in some precise sense. It remained an open question whether the soundness of this test can be pushed all the way down to $\exp(-k)$ (which would be optimal). This is our main result: we show that whenever f passes the Z test with probability $\epsilon > \exp(-k)$, there must be a global reason for this: namely, f must be close to a product function on some $\Omega(\epsilon)$ fraction of its domain.

Towards proving our result we analyze the related (two-query) V-test, and prove a "restricted global structure" theorem for it. Such theorems were also proven in previous works on direct product testing in the small soundness regime. The most recent work, by Dinur and Steurer (CCC 2014), analyzed the V test in the exponentially small soundness regime. We strengthen their conclusion of that theorem by moving from an "in expectation" statement to a stronger "concentration of measure" type of statement, which we prove using hyper-contractivity. This stronger statement allows us to proceed to analyze the Z test.

We analyze two variants of direct product tests. One for functions on ordered tuples, as above, and another for functions on sets, $f : \binom{[N]}{k} \to [M]^k$. The work of Impagliazzo et. al was actually focused only on functions of the latter type, i.e. on sets. We prove exponentially small soundness for the Z-test for both variants. Although the two appear very similar, the analysis for tuples is more tricky and requires some additional ideas.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Direct Product Testing, Property Testing, Agreement

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.29

1 Introduction

A function $f : [N]^k \to [M]^k$ for $N, M, k \in \mathbb{N}$, is a *direct product function* if $f = (g_1, \ldots g_k)$, for $g_i : [N] \to [M]$, i.e. the output of f on each coordinate depends on the input to this coordinate alone. Direct products appear in a variety of contexts in complexity, usually for hardness amplification. In PCPs it underlies the parallel repetition theorem [12] and implicitly appears in other forms of gap amplification, e.g. [4]. The specific task of testing

* Research supported in part by an ISF-UGC grant number 1399/14, and by BSF grant number 2014371.

direct products as an abstraction of a certain element of PCP constructions was introduced by [8].

The combinatorial question that underlies these works is the direct product testing question: given a function $f : [N]^k \to [M]^k$, is it a direct product function? The setting of interest here is where we query f in the *smallest number of inputs possible*, and decide if is it a direct product function or not.

The direct product testing question is a type of property testing question, yet it is not in the standard property testing parameter regime. In property testing we are generally interested in showing that functions that pass the test with high probability, for example 99%, are close to having the property.

In our case, we are interested in understanding the structure of functions that pass the test with small – but non-trivial – probability, e.g. 1%. The 1% regime is often more challenging than the 99% regime. It plays an important role in PCPs where one needs to prove a large gap. In such arguments one needs to be able to deduce non trivial structure even from a proof that passes a verification test with small probability, e.g. 1%.

There are very few families of tests for which 1% theorems are known. These include algebraic low degree tests and direct product tests. For low degree tests there has been a considerable amount of work in various regimes and in particular towards understanding the extent of the 1% theorems, see e.g. [13, 1, 3] and [2]. It is intriguing to understand more broadly for which tests such theorems can hold. Indeed, as far as we know, there are no other tests that exhibit such strong "structure vs. randomness" behavior, and direct product tests are natural candidates in which to study this question.

We remark that finding new settings where 1% theorems hold (including in particular derandomized direct products) can be potentially useful for constructing locally testable codes and stronger PCPs, see e.g. the recent works of [10, 6]. Towards this goal gaining a more comprehensive understanding of direct product tests, as well as developing tools for proving them, is a natural goal.

1.1 Our Main Result

The main question we study is: if $f : [N]^k \to [M]^k$ passes a certain natural test (Test 1 below) with non-negligible probability, how can f look like? We prove

▶ **Theorem 1** (Main Theorem – Global Structure). *For every $N, M > 1$, there exist small constants $c_1, c_2 > 0$ such that for every constant $\lambda > 0$ and large enough k, if $f : [N]^k \to [M]^k$ is a function that passes Test 1 with probability $\alpha_{Z(\frac{k}{10})}(f) = \epsilon \geq e^{-c_1 \lambda^2 k}$, then there exist functions $(g_1, \ldots g_k)$, $g_i : [N] \to [M]$ such that*

$$\Pr_{x \in [N]^k} \left[f(x) \overset{\lambda k}{\approx} (g_1(x_1) \ldots g_k(x_k)) \right] \geq c_2 \cdot \epsilon.$$

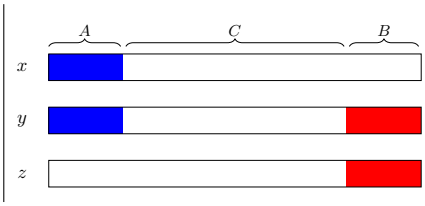
Where $\overset{\lambda k}{\approx}$ means that the strings are equal on all but at most λk coordinates.

The theorem is qualitatively tight with respect to several parameters: (i) Soundness, (i.e. the parameter ϵ), (ii) Approximate equality vs. exact equality (i.e. the parameter λ), (iii) Number of queries in the test. We discuss these next.

(i) Soundness

The soundness of the theorem is the smallest success probability in which the theorem is valid, in our case it is 2^{-ck} for some constant $c > 0$. This is tight up to the constant c, as can be seen by the example bellow.

1. Choose A, B, C to be a random partition of $[k]$, such that $|A| = |B| = t$.
2. Choose uniformly at random $x, y, z \in [N]^k$ such that $x_A = y_A$ and $y_B = z_B$.
3. Reject if $f(x)_A \neq f(y)_A$ or $f(z)_B \neq f(y)_B$, else accept.



Denote by $\alpha_{Z(t)}(f)$ the success probability of f on this test.

Test 1 "Z"-test with parameter t (3-query test).

▶ **Example 2** (Random function)**.** Let $f : [N]^k \to \{0,1\}^k$ be a random function; i.e. for each $x \in [N]^k$ choose $f(x) \in \{0,1\}^k$ uniformly and independently. Two random strings in $\{0,1\}^t$ are equal with probability 2^{-t}, therefore $\alpha_{Z(t)}(f) = 2^{-2t}$, since the test performs two such checks. On the other hand, since f is random, it is not close to any direct product function.

We remark that every function $f : [N]^k \to \{0,1\}^k$ is at least 2^{-k} close to a direct product function [1], so this amount of correlation is meaningless. We conclude that in order to have direct product theorem that is not trivial, the minimal soundness has to be $2^{-c'k}$ for some constant $c' < 1$.

(ii) Approximate equality vs. exact equality

In the theorem, we prove that for $\Omega(\epsilon)$ of the inputs x: $f(x) \overset{\lambda k}{\approx} (g_1(x), \ldots, g_k(x))$. A priori, one could hope for a stronger conclusion in which $f(x) = (g_1(x), \ldots, g_k(x))$ for $\Omega(\epsilon)$ of the x's. However, Example 3 shows that for $t = \frac{k}{10}$, approximate equality is necessary.

▶ **Example 3** (Noisy direct product function)**.** This example is from [5]. Let f be a direct product function, except that on each input x we "corrupt" $f(x)$ on λk random coordinates by changing $f(x)$ on these coordinates into random values. For $\lambda < \frac{1}{10}$, the probability that Test 1 on f missed all the corrupted coordinates is $2^{-\Omega(\lambda k)}$, in which case the test succeeds. Since we have changed $f(x)$ on λk coordinates into random values, no direct product function can approximate f on more than $(1 - \lambda)$ of the coordinates.

From this example we conclude that for f that passes Test 1 for $t = \frac{k}{10}$ with probability $e^{-\delta\lambda k}$, it is not possible to approximate f on more than $(1 - \lambda)$ of the coordinates. Further discussion and examples for different intersection sizes (i.e. t) are in Section 6.

(iii) Number of queries in the test

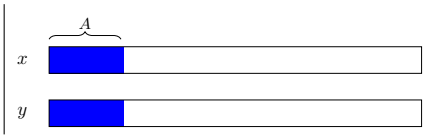
The absolute minimal number of queries for any direct product test is two. Indeed, there is a very natural 2-query test, Test 2.

Dinur and Goldenberg showed that it is not possible to have a direct product theorem with soundness lower than $\frac{1}{poly(k)}$ using the 2-query test [5].

▶ **Example 4** (Localized direct product functions)**.** In this example we assume $N \gg k$. For every $b \in [N]$ we choose a random function $g_b : [N] \to [M]$ independently. For every input $x \in [N]^k$, we choose a random $i_x \in k$, set $b = x_i$ and set $f(x) = (g_b(x_1), \ldots, g_b(x_k))$.

[1] Consider the direct product function constructed incrementally by taking the most common value out of $\{0,1\}$ on each step.

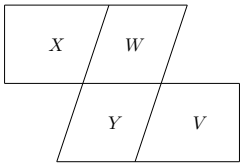
1. Choose $A \subset [k]$ of size t, uniformly at random.
2. Choose uniformly at random $x, y \in [N]^k$ such that $x_A = y_A$.
3. Accept if $f(x)_A = f(y)_A$.



Denote by $\alpha_{V(t)}(f)$ the success probability of f on this test.

Test 2 "V" test with parameter t (2-query test).

1. Choose random $V, W, X, Y \subset [N]$, such that $|W| = |V| = t$, $|X| = |Y| = k - t$ and $X \cap W = Y \cap W = Y \cap V = \emptyset$.
2. Reject if $f(X \cup W)_W \neq f(Y \cup W)_W$ or $f(Y \cup W)_Y \neq f(Y \cup V)_Y$, else accept.



Denote by $\alpha_{Z_{set}(t)}(f)$ the success probability of f on this test.

Test 3 "Z" test for functions over sets, with parameter t (3-queries).

The function f satisfies $\alpha_{V(t)}(f) \geq \frac{1}{k} \cdot \frac{t}{k}$; indeed, for x, y and A chosen in the test, if $i_x = i_y$ and $i_x \in A$, then the test will pass. The probability that $i_x = i_y$ is $\frac{1}{k}$, and the probability that $i_x \in A$ is $\frac{t}{k}$.

For $N \gg k$, the function f is very far from direct product, since it is made up from N different direct product functions. Each piece consisting of roughly $1/N$ fraction of the domain $[N]^k$.

For every t, the function described in the example satisfies $\alpha_{V(t)}(f) \geq \frac{1}{k^2}$, yet there is no direct product function that approximates f when $N \gg k$. In [5] the conclusion from Example 4 was that $1/\operatorname{poly}(k)$ is the limit for small soundness for direct product tests. However, [9] showed that by adding just one more query, this limitation goes away. They introduced a 3-query test, similar to Test 1, and proved a direct product theorem for all $\epsilon > 2^{-k^\beta}$ for some constant $\beta \leq 1/2$.

Direct product test for functions over sets

Some of the previous direct product works, such as [9] were proven in a slightly different setting, where the function tested is $f : \binom{[N]}{k} \to [M]^k$, i.e. the input to the function f is an unordered set $S \subset [N]$ of k elements. In this work, we also prove a direct product testing theorem for this setting, Test 3 is the analog of Test 1 for functions over sets. In Test 3 (see figure), we pick disjoint sets W, X, Y, Z such that $X \cap W = Y \cap W = Y \cap V = \emptyset$ so that $|X \cup W| = |Y \cup W| = |Y \cup V| = k$ and they can be inputs to the function f.

▶ **Theorem 5** (Global Structure for Sets). *There exist a small constant $c > 0$, such that for every constant $\lambda > 0$, large enough $k \in \mathbb{N}$ and $N > k^2 e^{10c\lambda k}$, if the function $f : \binom{[N]}{k} \to [M]^k$ passes Test 3 with probability $\alpha_{Z_{set}(\frac{k}{10})}(f) = \epsilon > e^{-c\lambda k}$, then there exist a function $g : [N] \to [M]$ such that*

$$\Pr_S \left[f(S) \overset{\lambda k}{\approx} g(S) \right] \geq \epsilon - 4\epsilon^2 .$$

Notice that the probability bound of $\epsilon - 4\epsilon^2$ is better than $\Omega(\epsilon)$, and it is tight as demonstrated by the function f which is a hybrid of $\frac{1}{\epsilon}$ different direct product functions on equals parts of

the inputs. f passes Test 3 with probability ϵ, and every direct product function is close to f only on ϵ fraction of the inputs.

We remark that the two theorems are not the same. In Theorem 1, there are k different functions $g_1, \ldots, g_k : [N] \to [M]$ whereas in Theorem 5 there is a single one. Furthermore, Theorem 1 holds for any $N, M \in \mathbb{N}$ and large enough k, and Theorem 5 (and other such direct product theorems) only holds for $N \gg k$. The proofs of the theorems are also different, which is discussed later in the introduction.

1.2 Restricted Global Structure

Our proof has two main parts, similar to the structure of the proof of [5, 9]. In the first part, we analyze only Test 2 (which is on tuples) and prove a restricted global structure theorem for it, Theorem 6 below (this was called local structure in [9, 7]). The term "restricted global structure" refers to when we restrict the domain to small (but not trivial) pieces, and show that f is close to a product function on each piece separately. This is the structure of the function in Example 4.

More explicitly, for every $A \in [k]$ of size $\frac{k}{10}$, $r \in [N]^A$ and $\gamma \in [M]^A$, a *restriction* is a triple $\tau = (A, r, \gamma)$. The choice of $t = \frac{k}{10}$ in Theorem 1 is somewhat arbitrary, the theorem can be proven with $t = ck$ for $c < \frac{1}{2}$. The restriction corresponds to the set of inputs

$$\mathcal{V}_\tau = \{w \in [N]^{[k]\setminus A} | f(r, w)_A = \gamma\}.$$

Our next theorem shows that for many restrictions τ there exist a direct product function that is close to f on $\mathcal{V}_\tau$.

▶ **Theorem 6** (Restricted Global Structure – informal). *Let $f : [N]^k \to [M]^k$ be a function that passes Test 2 with probability $\alpha_{V(\frac{k}{10})}(f) = \epsilon > e^{-\delta\lambda k}$, then there exist a natural distribution over restrictions $\tau = (A, r, \gamma)$ such that with probability $\Omega(\epsilon)$, there exist functions $(g_1^\tau, \ldots g_{\frac{9k}{10}}^\tau), g_i^\tau : [N] \to [M]$ such that,*

$$\Pr_{w \in [N]^{[k]\setminus A}} \left[f(r, w)_{[k]\setminus A} \overset{\lambda k}{\approx} (g_1^\tau(w_1), \ldots g_{\frac{9k}{10}}^\tau(w_{\frac{9k}{10}})) \;\middle|\; w \in \mathcal{V}_\tau \right] \geq 1 - \epsilon^2. \tag{1}$$

Where the distribution over τ is the test distribution, namely choose $A \subset [k], x \in [N]^k$ uniformly, and set $\tau = (A, x_A, f(x)_A)$.

A similar theorem was proven in [9] but only for soundness (i.e. ϵ) at least $\exp(-k^\beta)$ for a constant $\beta \leq 1/2$. This was strengthened to soundness $\exp(-\Omega(k))$ in [7]. Our Theorem 6 improves on the conclusion of [7] . In [7] the probability in (1) was shown to be at least $1 - O(\lambda)$ (recall that λ is a constant), whereas we show it is exponentially close to 1 (when ϵ is that small). This difference may seem minor but in fact it is what prevented [7] from deriving global structure via a three query test (i.e. moving from the V test to the Z test). When we try to move from restricted global structure to global structure, the consistency inside each restriction needs to be very high for the probabilistic arguments to work, as we try to explain below.

The restricted global structure gives us a direct product function that approximates f only on a restricted subset of the inputs. In the proof of the global structure, we use the third query to show that there exists a global function. A key step in the proof of the global structure is to show that for many restrictions τ, the function g^τ is close to f on a much larger subsets of inputs. This is done, intuitively, by claiming that if $f(x)_A = f(y)_A$, then with high probability $f(y) \approx g^\tau(y)$ for $\tau = (A, x_A, f(x)_A)$. Since B is a random set

and $f(z)_B = f(y)_B$, then $f(z), g^\tau(z)$ are also close. This claim only holds if the success probability on (1) is more than $1-\epsilon$, else it is possible that all the success probability of the test comes from f such that $f(x)_A = f(y)_A$, but $f(y), g^\tau(y)$ are far.

1.3 Technical Contribution

In terms of technical contribution our proof consists of two new components.

Domain extension

Our first contribution a new *domain extension* step that facilitates the proof of the restricted global structure. The restricted global structure shows that with probability $\Omega(\epsilon)$, the function f is close to a direct product on the restricted domain $\mathcal{V}_\tau$. A natural way to show that a function is close to a direct product function is to define a direct product function by majority value. However, this method fails when the agreement guaranteed for f is small, as in our case.

This is usually resolved by moving to a restricted domain in which the agreement is much higher, and to define majority there. The first part of our proof is to show that with probability $\Omega(\epsilon)$, over restrictions $\tau = (A, r, \gamma) \sim \mathcal{D}$, the set $\mathcal{V}_\tau$ satisfies the following two properties:

1. Its density is at least $\frac{\epsilon}{2}$.
2. f has very high agreement in $\mathcal{V}_\tau$, informally it means that taking a random pair $w, v \in \mathcal{V}_\tau$ such that $w_J = v_J$, results in agreeing answers, i.e. $f(r,w)_J \approx f(r,v))_J$, with probability greater than $1-\epsilon^{120}$.

We call such restrictions excellent, following [9].

We show that for every excellent restriction $\mathcal{V}_\tau$, the restriction h_τ of f to $\mathcal{V}_\tau$, defined by $h_\tau(w) = f(r,w)_{[k]\setminus A}$, is close to a direct product function. The function h_τ has high agreement, which is good for defining majority, but unfortunately the low density of $\mathcal{V}_\tau$, which can be as low as $\frac{\epsilon}{2}$, which is exponentially small, is where the techniques used in [9] break down. In order to prove that h_τ is close to a direct product function, we use a local averaging operator to *extend* the domain from $\mathcal{V}_\tau$ to $[N]^{[k]\setminus A}$.

The local averaging operator $\mathcal{P}_{\frac{3}{4}}$ is the majority of a $\frac{3}{4}$-correlated neighborhood,

$$\forall w \in [N]^{[k]\setminus A}, i \notin A \qquad \mathcal{P}_{\frac{3}{4}} h_\tau(w)_i = \underset{v \in \mathcal{N}_{\frac{3}{4}}(w), v \in \mathcal{V}_\tau, v_i = w_i}{\text{Plurality}} \{h_\tau(v)_i\},$$

where $v \in \mathcal{N}_{\frac{3}{4}}(w)$ means that v is $\frac{3}{4}$-correlated with w, i.e. we change each coordinate of w with probability $\frac{1}{4}$ independently. The new function, $\mathcal{P}_{\frac{3}{4}} h_\tau$ is defined over all $[N]^{[k]\setminus A}$, unlike h_τ which is defined only on $\mathcal{V}_\tau$.

In order to use $\mathcal{P}_{\frac{3}{4}} h_\tau$ for showing that h_τ is close to a direct product function, we show two things:

1. $\mathcal{P}_{\frac{3}{4}} h_\tau$ and h_τ are similar on $\mathcal{V}_\tau$.
2. $\mathcal{P}_{\frac{3}{4}} h_\tau$ has high agreement, taking a random pair $w, v \in [N]^{[k]\setminus A}$ such that $w_J = v_J$, results in agreeing answers, $\mathcal{P}_{\frac{3}{4}} h_\tau(w)_J \approx \mathcal{P}_{\frac{3}{4}} h_\tau(v)_J$ with probability $1-\epsilon^6$.

To prove that $\mathcal{P}_{\frac{3}{4}} h_\tau$ has high agreement we use reverse hypercontractivity to show that only a few $w \in [N]^{[k]\setminus A}$ have sparse neighborhood (with density less than ϵ^{50}), and use the very high agreement of h_τ.

Lastly, we define a direct product function g_τ by taking the plurality over $\mathcal{P}_{\frac{3}{4}} h_\tau$, and show that it is close to h_τ.

Direct product testing in a dense regime

A second new element comes when stitching the many localized functions into one global direct product function, by using the third query.

We prove two global structure theorems, Theorem 1 for functions on tuples $f : [N]^k \to [M]^k$ and Theorem 5 for functions on sets $f : \binom{[N]}{k} \to [M]^k$.

When we work with f that is defined over sets, we can directly follow the approach of [9] to complete the proof. However, when working with f defined on tuples we reach a combinatorial question that itself resembles a direct product testing question, but in a different (dense) regime. Luckily, the fact that this question is in a dense regime makes it easier to solve, and this leads to our global structure theorem for tuples. An outline of the global structure proofs appears in Section 5.1.

1.4 Agreement Tests and Direct Product Tests

The question of direct product testing fits into a more general family of tests called agreement tests. We next describe this setting formally and explain how direct product tests fit into this framework.

Agreement tests

In all efficient PCPs we break a proof into small overlapping pieces, use a relatively inefficient PCPs (i.e. PCPs that incur a large blowup) to encode each small piece, and then through an *agreement test* put the pieces back together. The agreement test is needed because given the collection of pieces, there is no guarantee that the different pieces come from the same underlying global proof, i.e. that the proofs of each piece can be "put back together again". The PCP system needs to ensure this through *agreement testing*: we take two pieces that have some overlap, and check that they agree.

This situation can be formulated as an agreement testing question as follows. Let V be a ground set, $|V| = N$, and let H be a collection of subsets of V, namely, a set of hyperedges. Let $[M]$ be a finite set of colors, where it is sufficient to think of $M = 2$.

A *local assignment* is a collection $a = \{a_s\}$ of local colorings $a_s : s \to [M]$, one per subset $s \in H$. A local assignment is called *global* if there is a global coloring $g : V \to [M]$ such that

$$\forall s \in H, \qquad a_s \equiv g|_s.$$

An *agreement check* for a pair of subsets s_1, s_2 checks whether their local functions agree, denoted $a_{s_1} \sim a_{s_2}$. Formally,

$$a_{s_1} \sim a_{s_2} \qquad \Leftrightarrow \qquad \forall x \in s_1 \cap s_2, \quad a_{s_1}(x) = a_{s_2}(x).$$

A local assignment that is global passes all agreement checks. The converse is also true: a local assignment that passes *all* agreement checks must be global.

An *agreement test* is specified by giving a distribution $\mathcal{D}$ over pairs (or triples) of subsets s_1, s_2. We define the agreement of a local assignment to be the probability of agreement,

$$agree_{\mathcal{D}}(a) = \Pr_{s_1,s_2 \sim D} [a_{s_1} \sim a_{s_2}] .$$

An agreement theorem shows that if a is a local assignment with $agree_{\mathcal{D}}(a) > \epsilon$ then a is somewhat close to a global assignment. Agreement theorems can be studied for any hypergraph and in this work we prove such theorems for two specific hypergraphs: the k-uniform complete hypergraph, and the k-uniform k-partite complete hypergraph.

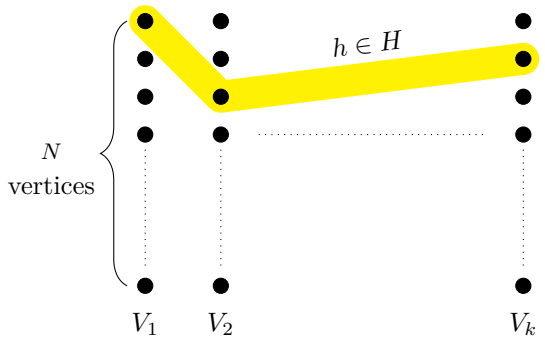


Figure 1 Complete k-uniform k-partite graph.

Relation to direct product testing

Theorem 1 is equivalent to an agreement theorem on the *complete k-uniform k-partite hypergraph* (see Figure 1). Let $G = (V = V_1, \dots V_k, H)$ be the complete k-partite hypergraph with $|V_i| = N$ for $i \in [k]$, and

$$H = \{(v_1, \dots v_k) \mid \ \forall i \in [k], v_i \in V_i\}\,.$$

There is a bijection between H and $[N]^k$. We shall interpret $f(x_1, \dots, x_k)$ as a local coloring of the vertices $x_1, \dots, x_k$. In this way, we have the following equivalence

$$f : [N]^k \to [M]^k \qquad \iff \qquad a = \{a_x\}_{x \in H}.$$

Moreover, local assignments which are global, i.e. a such that $a_x = g|_x$ for some global coloring $g : V_1 \cup \dots \cup V_k \to [M]$, correspond exactly to functions f which are direct products, $f = (g_1, \dots, g_k)$ where $g_i = g|_{V_i}$,

$$f = (g_1, \dots, g_k) \qquad \iff \qquad a \text{ is global.}$$

Finally, Test 2 can be described as taking 2 hyperedges that intersect on t vertices, and check if their local functions agree on the intersection. Similarly, Test 1 can be described as picking three hyperedges, $h_1, h_2, h_3 \in H$ such that h_1, h_2 intersect on t vertices, and h_2, h_3 intersect on a disjoint set of t vertices, and checking agreement.

Our main theorem, Theorem 1, is equivalent to an agreement theorem showing that if a local assignment a passes a certain 3-query agreement test with non-negligible probability, then there exists a global assignment $g : V \to [M]$ with which it agrees non-negligibly.

The k-uniform complete hypergraph (it is non-partite, in contrast to the above), is related to Theorem 5. In this hypergraph the vertex set is $[N]$ and there is a hyperedge for every possible k-element subset of $[N]$. Now we have a similar equivalence between local assignments and functions over sets, i.e. functions where the input is a set $S \subset [N]$ of size k,

$$f : \binom{[N]}{k} \to [M]^k \qquad \iff \qquad a = \{a_s\}_{s \in \binom{[N]}{k}}.$$

An agreement theorem for this hypergraph is equivalent to Theorem 5, in which f is defined not on "tuples" $[N]^k$ but on "sets" $\binom{[N]}{k}$. A global assignment a or this graph is equivalent to a direct product function over sets, i.e. $f = g : [N] \to [M]$.

1.5 Organization of the Paper

Section 2 contains preliminary notations and definitions. In Section 3 we prove the restricted global structure, Theorem 6. Section 4 is dedicated to the global structure for functions on sets. We show how to deduce a variant of Theorem 6 for sets rather than tuples and then prove the global structure theorem for sets, Theorem 5. In Section 5 we prove the global structure theorem for tuples, Theorem 1. Lastly, in Section 6 we discuss lower bounds for various 3-query direct product tests that were not presented in the introduction.

2 Preliminaries

▶ **Definition 7.** For each two strings $x, y \in [N]^k$ we say that:

1. $x \overset{t}{\approx} y$ if x, y differ in at most t coordinates.
2. $x \overset{t}{\not\approx} y$ if x, y differ in more than t coordinates.

▶ **Definition 8** (Plurality)**.** The plurality of a function f on a distribution $\mathcal{D}$ is its most frequent value

$$\underset{x\sim\mathcal{D}}{\text{Plurality}}(f(x)) = \arg\max_{\beta}\left\{\Pr_{x\in\mathcal{D}}[f(x)=\beta]\right\}$$

For a set $A \subset [k]$ we denote by $\bar{A}$ the set $[k] \setminus A$.

▶ **Fact 9** (Chernoff bound)**.** *Let $X_1, \ldots X_k$ be independent random variables in $\{0,1\}$, let $X = \sum_{i=1}^{k}$, and denote $\mu = \mathbb{E}[X]$, then for every $\delta \in (0,1)$,*

$$\Pr_{X_1,\ldots X_k}[X \leq (1-\delta)\nu] \leq e^{-\frac{\delta^2\mu^2}{2}},$$

and for every $\delta \in (0,1]$

$$\Pr_{X_1,\ldots X_k}[X \geq (1+\delta)\nu] \leq e^{-\frac{\delta^2\mu^2}{3}}.$$

▶ **Corollary 10.** *Let k be a large integer, and let $A \subseteq [k]$ be the set generated by inserting each $i \in [k]$ into A with probability ρ. For every constant $c \in (0,1)$*

$$\Pr_A[|A| \leq c\rho k] \leq e^{-\frac{(1-c)^2}{2}\rho k},$$

and for every $c' \in [1,2]$,

$$\Pr_A[|A| \geq c'\rho k] \leq e^{-\frac{(c'-1)^2}{3}\rho k}.$$

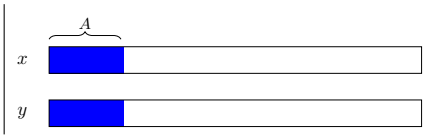
▶ **Claim 11** (Chernoff bound for fixed size subsets)**.** *Let $k \in \mathbb{N}$ be a large integer, $D \subset [k]$ be a fixed subset of size at most $\frac{k}{3}$. Let A be a random subset of size exactly $\frac{k}{10}$, then*

$$\Pr_A\left[|A\cap D| \geq \frac{1}{5}|D|\right] \leq e^{-\frac{1}{320}|D|} \tag{2}$$

If $|D| \leq \frac{1}{30}k$ then

$$\Pr_A\left[|A\cap D| \leq \frac{1}{20}|D|\right] \leq e^{-\frac{1}{60}|D|} \tag{3}$$

1. Choose $A \subset [k]$ of size t, uniformly at random.
2. Choose uniformly at random $x, y \in [N]^k$ such that $x_A = y_A$.
3. Accept if $f(x)_A = f(y)_A$.



Denote by $\alpha_{V(t)}(f)$ the success probability of f on this test.

Test 2 "V" test with parameter t (2-query test).

The proof appears in Appendix A.

In our proof we also need Chernoff bound for non-binary random variables.

▶ **Fact 12** (Non-binary Chernoff bound). *Let $X_1, \ldots X_k$ be independent random variables in $[0,1]$, let $X = \sum_{i=1}^{k} X_i$, and denote $\mu = \mathbb{E}[X]$ then,*

$$\Pr_{X_1,\ldots X_k}[|X - \mu| > t] \leq 2e^{-t^2 k},$$

2.1 Reverse Hypercontractivity

▶ **Definition 13** (ρ-correlated distribution). For each string $y \in [N]^k$ and constant $\rho \in (0,1)$, the ρ correlated distribution from y will be denoted by $(x, J) \in \mathcal{N}_\rho(y)$. For each $i \in [k]$ independently, $i \in J$ with probability ρ, and x is chosen such that $x_J = y_J$, and the rest is uniform.

We quote Proposition 9.2 from [11]:

▶ **Claim 14.** *Let $A, B \subseteq [N]^k$ of sizes $\Pr_{w\in[N]^k}[w \in A] = e^{-\frac{a^2}{2}}$ and $\Pr_{w\in[N]^k}[w \in B] = e^{-\frac{b^2}{2}}$, then*

$$\Pr_{x\in[N]^k, y\in\mathcal{N}_\rho(x)}[x \in A, y \in B] \geq e^{-\frac{(2-\rho)(a^2+b^2)}{4(1-\rho)} - \frac{\rho ab}{2(1-\rho)}}.$$

By changing notations and simplifying, we get the following corollary.

▶ **Corollary 15.** *For $|A| \geq |B|$,*

$$\Pr_{x\in[N]^k, y\in\mathcal{N}_\rho(x)}[x \in A, y \in B] \geq \Pr_{x\in[N]^k}[x \in A]^{1+\frac{\rho}{2(1-\rho)}} \Pr_{x\in[N]^k}[x \in B]^{1+\frac{3\rho}{2(1-\rho)}}.$$

▶ **Claim 16.** *Let $G \subset [N]^k$ be a set of measure ν, then for any $\eta \in (0,1)$ the set $L = \left\{ w \in [N]^k \;\middle|\; \Pr_{(v,J)\in\mathcal{N}_{\frac{3}{4}}(w)}[v \in G] \leq \eta \right\}$ has a measure less than $\nu^{-\frac{11}{9}}\eta^{\frac{2}{9}}$.*

Both proofs appears in Appendix A.

3 Restricted Global Structure

Let $f : [N]^k \to [M]^k$ be such that $\alpha_{V(\frac{k}{10})}(f) = \epsilon \geq e^{-c\lambda k}$, i.e. the success probability of f on Test 2 equals ϵ. To make the reading easy, we write again Test 2 from the introduction.

We show in this section that $\alpha_{V(\frac{k}{10})}(f) = \epsilon$ already implies that f is somewhat structured, namely there are restrictions of the domain $\mathcal{V}_\tau \subset [N]^k$ such that on these restrictions f is roughly a product function.

Recalling the definition from the introduction, we define a *restriction* to be a triple $\tau = (A, r, \gamma)$, for $A \subset [k]$, $r \in [N]^A$ and $\gamma \in [M]^A$. In this section denote by $k' = \frac{9k}{10}$, and recall that $\bar{A} = [k] \setminus A$.

▶ **Definition 17** (Consistent strings)**.** For each restriction $\tau = (A, r, \gamma)$, a string $w \in [N]^{\bar{A}}$ is consistent with τ if $f(r, w)_A = \gamma$. For every τ, let $\mathcal{V}_\tau$ be the set of consistent strings,

$$\mathcal{V}_\tau = \left\{ w \in [N]^{\bar{A}} \;\middle|\; f(r, w)_A = \gamma \right\}.$$

▶ **Definition 18** (Distribution of Restrictions)**.** Let $\mathcal{D}$ be the following distribution over restrictions τ. Pick a uniform set $A \subset [k]$ of size $\frac{k}{10}$, pick a uniform $x \in [N]^k$ and set $r = x_A$ and $\gamma = f(x)_A$.

Note that the distribution $\mathcal{D}$ depends on the function f.

We define good restriction in an analogous way to the definitions of [9].

▶ **Definition 19** (Good restriction)**.** A restriction $\tau = (A, r, \gamma)$ is *good*, if $\Pr_{w \in [N]^{\bar{A}}}[w \in \mathcal{V}_\tau] \geq \frac{\epsilon}{2}$.

▶ **Definition 20** (DP restriction)**.** A restriction $\tau = (A, r, \gamma)$ is a *DP restriction* if it is good, and if there exist functions $(g_1^\tau, \ldots g_{k'}^\tau), g_i^\tau : [N] \to [M]$ such that

$$\Pr_{w \in [N]^{\bar{A}}} \left[f(r, w)_{\bar{A}} \overset{\lambda k}{\not\approx} (g_1^\tau(w_1), \ldots g_{k'}^\tau(w_{k'})) \;\middle|\; w \in \mathcal{V}_\tau \right] \leq \epsilon^2.$$

The main theorem of this section shows that (a) a non-negligible fraction of restrictions are good, and that (b) almost all good restrictions are DP restrictions.

▶ **Theorem 21** (Restricted Global Structure, restated)**.** *There exist a small constant $\delta > 0$, such that for every constant $\lambda > 0$ and large enough $k \in \mathbb{N}$ the following holds. For every function $f : [N]^k \to [M]^k$, if $\alpha_{V(\frac{k}{10})}(f) = \epsilon > e^{-\delta \lambda k}$, then with probability at least $\frac{\epsilon}{2}$, $\tau \sim \mathcal{D}$ is good, and with probability at least $1 - \epsilon^2$ over the good restrictions, τ is a DP restriction. Namely, τ is such that there exist functions $(g_1^\tau, \ldots g_{k'}^\tau), g_i^\tau : [N] \to [M]$ such that*

$$\Pr_{w \in [N]^{\bar{A}}} \left[f(r, w)_{\bar{A}} \overset{\lambda k}{\not\approx} (g_1^\tau(w_1), \ldots g_{k'}^\tau(w_{k'})) \;\middle|\; w \in \mathcal{V}_\tau \right] \leq \epsilon^2.$$

A similar theorem was proven in [7] under the name "local structure". Under the same assumptions [7] showed that f must be close to a product function for many restrictions $\mathcal{V}_\tau$ of the domain. However the closeness was considerably weaker: unlike in our definition of a *DP restriction*, in [7] even in the restricted part of the domain, $\mathcal{V}_\tau \subset [N]^k$, there could be a (small) constant fraction of the inputs on which f differs from the global product function g^τ. In contrast, we only allow an ϵ^2 fraction of disagreeing inputs. As explained in the introduction, in order to extend the restricted global structure into a global one, the set of disagreeing inputs in $\mathcal{V}_\tau$ has to be smaller than ϵ.

3.1 Proof of Theorem 21

In this section we prove Theorem 21, we start by writing a few definitions and lemmas that are used in the proof, and give an intuition for the proof of each lemma. We defer the proofs of these lemmas to the next sections.

The distribution $\mathcal{D}$ over τ is related to the distribution of Test 2. The test can also be written as choose $\tau = (A, r, \gamma) \sim \mathcal{D}$, $w \in [N]^{\bar{A}}$ and accept iff $f(r, w)_A = \gamma$. Therefore, if the function f passes Test 2 with probability ϵ, by a simple averaging argument

$$\Pr_{\tau \sim \mathcal{D}} [\tau \text{ is good}] \geq \frac{\epsilon}{2}. \tag{4}$$

For each τ we define the function h_τ, which is a restriction of f to $\mathcal{V}_\tau$.

▶ **Definition 22.** For each restriction $\tau = (A, r, \gamma)$, let $h_\tau : \mathcal{V}_\tau \to [M]^{\frac{9k}{10}}$ be the function,

$$h_\tau(w) = f(r, w)_{\bar{A}}.$$

We define excellent restriction, in an analogous way to [9],

▶ **Definition 23** (Excellent restriction)**.** Fix a constant $\alpha = \frac{1}{1600}\lambda$, a restriction $\tau = (A, r, \gamma)$ is excellent, if:

1. τ is good.
2. For every $\rho \in \left\{ \frac{a}{b} \;\middle|\; a, b \in \mathbb{N}, a < b \leq k \right\}$, if we pick $w \in [N]^{\bar{A}}$ and $(v, J) \in \mathcal{N}_\rho(w)$ then,

$$\Pr_{w,(v,J)}\left[w, v \in \mathcal{V}_\tau,\ h_\tau(w)_J \overset{\alpha k}{\not\approx} h_\tau(v)_J \right] \leq \left(\frac{9}{10}\right)^{\frac{1}{2}\alpha k}. \tag{5}$$

Note that (5) holds trivially when $\rho < \alpha$, because with high probability $|J| \approx \rho k < \alpha k$, in which it is not possible that $h(w)_J, h(v)_J$ differs in more than αk coordinates. For an excellent τ, the set $\mathcal{V}_\tau$ is of measure at least $\frac{\epsilon}{2}$, and the function f is consistent on $\mathcal{V}_\tau$.

We assume that the constant δ is small enough to satisfy $\left(\frac{9}{10}\right)^{\frac{1}{2}\alpha k} < \epsilon^{120} = e^{-120\delta\lambda k}$, and $\epsilon^{120} > e^{-\frac{\alpha k}{43000}}$.

▶ **Lemma 26.** *For every $\rho \in (0, 1)$, let $\tau = (A, r, \gamma) \sim \mathcal{D}$, let $w \in [N]^{\bar{A}}$ be uniform, and let $(v, J) \in \mathcal{N}_\rho(w)$, then*

$$\Pr_{\tau,w,(v,J)}\left[w, v \in \mathcal{V}_\tau,\ h_\tau(w)_J \overset{\alpha k}{\not\approx} h_\tau(v)_J \right] \leq \left(\frac{9}{10}\right)^{\alpha k}.$$

The proof appears on Section 3.2, the main idea in the proof is that the probability of $w, v \in \mathcal{V}_\tau, h(w)_J \overset{\alpha k}{\not\approx} h(v)_J$ is low when averaging over τ as well. From the definition of h_τ, this is equivalent to $f(r, w)_A = f(r, v)_A = \gamma$ and $f(r, w)_J \overset{\alpha k}{\not\approx} f(r, v)_J$. When r, w, v, A, J are all random, the probability for a uniform A, J to be such that $f(r, w), f(r, v)$ are equal on A but far on J is very small.

▶ **Corollary 24.** *A good $\tau \sim \mathcal{D}$ is excellent with probability larger than $1 - \epsilon^2$.*

Proof. Let $\mu = \left(\frac{9}{10}\right)^{\frac{1}{2}\alpha k}$, and denote by $E(\tau, w, v, J)$ the event of $w, v \in \mathcal{V}_\tau$, $h_\tau(w)_J \overset{\alpha k}{\not\approx} h_\tau(v)_J$. Lemma 26 in these notations is: for every $\rho \in (0, 1)$, $\Pr_{\tau\sim\mathcal{D},w,(v,J)\in\mathcal{N}_\rho(w)}[E] \leq \mu^2$.

For every τ that is good but not excellent, exist $\rho \in \left\{ \frac{a}{b} \;\middle|\; a, b \in \mathbb{N}, a < b \leq k \right\}$ such that,

$$\Pr_{w,(v,J)\in\mathcal{N}_\rho(w)}[E] > \mu.$$

In this case we say that τ is bad for ρ.

Assume towards contradiction that $\Pr_{\tau\sim\mathcal{D}}[\tau \text{ is good but not excellent}] > \epsilon^4$. The set $\left\{ \frac{a}{b} \;\middle|\; a, b \in \mathbb{N}, a < b \leq k \right\}$ contains less than k^2 elements, so there exists ρ in this set such that

$$\Pr_{\tau\sim\mathcal{D}}[\tau \text{ is bad for } \rho] \geq \frac{\epsilon^4}{k^2}.$$

For this ρ,

$$\Pr_{\tau\sim\mathcal{D},w,(v,J)\in\mathcal{N}_\rho(w)}[E] \geq \Pr_{\tau\sim\mathcal{D}}[\tau \text{ is bad for } \rho] \Pr_{w,(v,J)\in\mathcal{N}_\rho(w)}[E \mid \tau \text{ is bad for } \rho] \geq \frac{\epsilon^3}{k^2}\mu.$$

This contradicts Lemma 26, because $\frac{\epsilon^4}{k^2}\mu \gg \mu^2$ (we assume that $\mu < \epsilon^{120}$). Therefore, we conclude that $\Pr_{\tau\sim\mathcal{D}}[\tau \text{ is good but not excellent}] \leq \epsilon^4$

Since $\tau \sim \mathcal{D}$ is good with probability at least $\frac{\epsilon}{2}$, by averaging a good $\tau \sim \mathcal{D}$ is excellent with probability at least $1-\epsilon^2$. ◀

In order to prove Theorem 21, it is enough to show that every excellent restriction is a *DP restriction*. A natural idea is to define a direct product function by taking the plurality of h_τ on $\mathcal{V}_\tau$, because the agreement of h_τ inside $\mathcal{V}_\tau$ is almost 1. However, it is difficult to prove that this function is close to h_τ because the set $\mathcal{V}_\tau$ is very sparse. We define a local averaging operator, which allows us to go from h_τ that is defined on $\mathcal{V}_\tau$, to a function that is defined on $[N]^{\bar{A}}$.

▶ **Definition 25** (Local averaging operator)**.** For every $\rho \in [0,1]$, let $\mathcal{P}_\rho$ be the following function operator. For every subset $\mathcal{V}_\tau \subset [N]^{\bar{A}}$, and every function $h : \mathcal{V}_\tau \to [M]^t$, the function $\mathcal{P}_\rho h : [N]^t \to [M]^t$ satisfies $\forall i \in [k], w \in [N]^t$,

$$\mathcal{P}_\rho h(w)_i = \underset{(v,J)\in\mathcal{N}_\rho(w), v_i=w_i}{\text{Plurality}} (h(v)_i).$$

If there is no v such that $v_i = w_i$ in $\mathcal{V}_\tau$, we define $\mathcal{P}_\rho h(w)_i$ to an arbitrary value.

The local averaging operator of h takes for every w and i the most frequent value $h(v)_i$ over a ρ-correlated neighborhood of w. We note that the function operator is not linear.

In order to prove that h_τ is close to a direct product function, we first show that that $\mathcal{P}_{\frac{3}{4}} h_\tau$ is close to h_τ, and then that $\mathcal{P}_{\frac{3}{4}} h_\tau$ is close to a direct product function. Clearly $\frac{3}{4}$ is an arbitrary constant, our proof works for any constant $\rho > \frac{1}{2}$, and we fix $\rho = \frac{3}{4}$.

▶ **Lemma 27.** *For every excellent τ,*

$$\Pr_{w\in[N]^{\frac{9k}{10}}}\left[h_\tau(w) \overset{12\alpha k}{\not\approx} \mathcal{P}_{\frac{3}{4}} h_\tau(w) \;\middle|\; w \in \mathcal{V}_\tau\right] \leq \epsilon^3.$$

The proof is in Section 3.3, and uses the very high consistency of h_τ inside $\mathcal{V}_\tau$ to show that the plurality vote is almost always consistent with $h_\tau(w)$. In the proof we use reverse hypercontractivity [11] to show that the set $\mathcal{V}_\tau$ is not too sparse, such that for almost all $w \in \mathcal{V}_\tau$, the neighborhood $\mathcal{N}_{\frac{3}{4}}(w)$ is not empty.

In a similar way to the proof of Lemma 27, we show that for an excellent τ the function $\mathcal{P}_{\frac{3}{4}} h_\tau$ has high agreement.

▶ **Lemma 28.** *For every excellent τ,*

$$\Pr_{w,(v,J)}\left[\mathcal{P}_{\frac{3}{4}} h(w)_J \overset{20\alpha k}{\approx} \mathcal{P}_{\frac{3}{4}} h(v)_J\right] \geq 1-\epsilon^{10},$$

where $w \in [N]^{\frac{9k}{10}}$ and $(v,J) \in \mathcal{N}_{\frac{1}{2}}(w)$.

The proof of this lemma also appears in Section 3.3, the main idea is that if $\mathcal{P}_{\frac{3}{4}} h(w_1), \mathcal{P}_{\frac{3}{4}} h(w_2)$ disagree on a lot of coordinates, then a large fraction of their $\frac{3}{4}$-correlated neighborhood also disagree on a lot of coordinates. This can only happen for very few inputs w, else we contradict the fact that τ is excellent.

After showing that $\mathcal{P}_{\frac{3}{4}} h_\tau$ has high agreement, we define g^τ to be the plurality vote of $\mathcal{P}_{\frac{3}{4}} h_\tau$, and then use the high agreement, Lemma 28, to show that they g^τ is close to $\mathcal{P}_{\frac{3}{4}} h_\tau$.

▶ **Lemma 29.** *For every excellent restriction τ there exist a direct product function $g^\tau = g_1^\tau \dots g_{\frac{9k}{10}}^\tau : [N]^{\frac{9k}{10}} \to [M]^{\frac{9k}{10}}$ such that*

$$\Pr_{w\in[N]^{\frac{9k}{10}}}\left[\mathcal{P}_{\frac{3}{4}}h_\tau(w) \overset{1500\alpha k}{\not\approx} g^\tau(w)\right] \le 3\epsilon^4.$$

The proof is in Section 3.4.

Using the above lemmas we can prove the local structure.

Proof of Theorem 21. Let $f : [N]^k \to [M]^k$ be a function that passes Test 2 with probability ϵ.

From averaging, $\Pr_{\tau\sim\mathcal{D}}[\tau \text{ is good}] \ge \frac{\epsilon}{2}$, Lemma 26 implies that with probability $(1-\epsilon^2)$, a good τ is also excellent.

Fix an excellent τ, by definition the function h_τ has high consistency inside $\mathcal{V}_\tau$, and by Lemma 27, $\mathcal{P}_{\frac{3}{4}}h$ is close to h on $\mathcal{V}_\tau$. Let $E_1(w)$ be the event that $h_\tau(w) \overset{12\alpha k}{\not\approx} \mathcal{P}_{\frac{3}{4}}h_\tau(w)$, in this notation Lemma 27 implies that

$$\Pr_w[E_1 \mid w \in \mathcal{V}_\tau] \le \epsilon^3. \tag{6}$$

From Lemma 29, there exists a product function g^τ that is similar to $\mathcal{P}_{\frac{3}{4}}h_\tau$. Denote by $E_2(w)$ the event that $\mathcal{P}_{\frac{3}{4}}h_\tau(w) \overset{1500\alpha k}{\not\approx} g^\tau(w)$. In this notation,

$$\Pr_w[E_2] \le 3\epsilon^4. \tag{7}$$

We want to use (6) and (7) to prove that h_τ is similar to g^τ on $\mathcal{V}_\tau$. In order to do that, we need to bound the probability of E_2 conditioned on $w \in \mathcal{V}_\tau$.

$$\begin{aligned}
3\epsilon^4 &\ge \Pr_w[E_2] \\
&\ge \Pr_w[w\in\mathcal{V}_\tau]\Pr_w[E_2 \mid w\in\mathcal{V}_\tau] && (\tau \text{ is excellent}) \\
&\ge \frac{\epsilon}{2}\Pr_w[E_2 \mid w\in\mathcal{V}_\tau].
\end{aligned}$$

Therefore $\Pr_w[E_2 \mid w \in \mathcal{V}_\tau] \le 6\epsilon^3$.

If w is such that none of E_1, E_2 happened, then $h_\tau(w), \mathcal{P}_{\frac{3}{4}}h_\tau(w)$ are equal in all but $12\alpha k$ of the coordinates, and $\mathcal{P}_{\frac{3}{4}}h_\tau(w), g^\tau(w)$ are equal in all but $1500\alpha k$ of the coordinates, which means that $h_\tau(w) \overset{1512\alpha k}{\approx} g^\tau(w)$.

$$\begin{aligned}
\Pr_w\left[h_\tau(w) \overset{1512\alpha k}{\not\approx} g^\tau(w) \;\middle|\; w\in\mathcal{V}_\tau\right] &\le \Pr_w[E_1 \vee E_2 \mid w\in\mathcal{V}_\tau] \\
&\le \Pr_w[E_1 \mid w\in\mathcal{V}_\tau] + \Pr_w[E_2 \mid w\in\mathcal{V}_\tau] \\
&\le \epsilon^3 + 6\epsilon^3 < \epsilon^2.
\end{aligned}$$

By definition, $h_\tau(w) = f(x_A, w)_{\bar{A}}$,

$$\Pr_w\left[f(x_A,w)_{\bar{A}} \overset{1512\alpha k}{\not\approx} g^\tau(w) \;\middle|\; w\in\mathcal{V}_\tau\right] = \Pr_w\left[h_\tau(w) \overset{1512\alpha k}{\not\approx} g^\tau(w) \;\middle|\; w\in\mathcal{V}_\tau\right] < \epsilon^2.$$

Since $\lambda = 1600\alpha$ we are done. ◀

3.2 Good Restrictions are Excellent with High Probability

For convenience, we restate the lemma.

▶ **Lemma 26.** *For every $\rho \in (0,1)$, let $\tau \sim \mathcal{D}$, $w \in [N]^{\frac{9k}{10}}$ and $(v,J) \in \mathcal{N}_\rho(w)$, then*

$$\Pr_{\tau,w,(v,J)}\left[w,v \in \mathcal{V}_\tau,\ h_\tau(w)_J \overset{\alpha k}{\not\approx} h_\tau(v)_J\right] \leq \left(\frac{9}{10}\right)^{\alpha k}.$$

Proof. Fix $\rho \in (0,1)$, let $E_1(\tau,w,v,J)$ be the event in equation (5) of the definition of excellence, Definition 23. More explicitly, $E_1 = 1$ if $w,v \in \mathcal{V}_\tau$ and $h_\tau(w)_J \overset{\alpha k}{\not\approx} h_\tau(v)_J$.

Recall the definition of h_τ for $\tau = (A,r,\gamma)$, for $w \in \mathcal{V}_\tau$, $h_\tau(w) = f(r,w)_{\bar{A}}$. Therefore, the event E_1 can also be written as $f(r,w)_A = f(r,v)_A = \gamma$ and $f(r,w)_{\bar{A}} \overset{\alpha k}{\not\approx} f(r,v)_{\bar{A}}$.

Let E_2 be the event that $f(r,w)_A = f(r,v)_A$ and $f(r,w)_{\bar{A}} \overset{\alpha k}{\not\approx} f(r,v)_{\bar{A}}$. We can easily see that $E_1 \subseteq E_2$, therefore over every distribution $\Pr[E_1] \leq \Pr[E_2]$.

We start by bounding the probability of event E_2, over the distribution $\tau \sim \mathcal{D}$, $w \in [N]^{\frac{9k}{10}}$ uniformly and $(v,J) \in \mathcal{N}_\rho(w)$. Writing the distribution explicitly:

1. Pick $A \subset [k]$ of size $\frac{k}{10}$.
2. Pick $x \in [N]^k$, set $r = x_A$ and $\gamma = f(x)_A$.
3. Pick $J \subset \left[\frac{9k}{10}\right]$ of size $B(\frac{9k}{10},\rho)$ (binomial random variable).
4. Pick uniform $w,v \in [N]^{\frac{9k}{10}}$ such that $w_J = v_J$.

Notice that E_2 is independent of γ, so it does not matter how γ is chosen. We can define an equivalent process for producing the same distribution (without γ):

1. Pick a set $A' \subset [k]$ of size $\frac{k}{10} + B(\frac{9k}{10},\rho)$.
2. Pick $y,z \in [N]^k$ such that $y_{A'} = z_{A'}$.
3. Pick $A \subseteq A'$ of size $\frac{k}{10}$.
4. Set $r = y_A$, $w = y_{\bar{A}}$ and $v = z_{\bar{A}}$.

In order of E_2 to happen, y,z,A' must be such that $f(y)_{A'} \overset{\alpha k}{\not\approx} f(z)_{A'}$. Furthermore, the set A must be chosen such that $f(y)_A = f(z)_A$. As the second random process allows us to see, A is a random subset of A', and each of the αk coordinates i on which $f(y)_i \neq f(z)_i$ has probability of at least $\frac{1}{10}$ to be chosen to A (as $|A| = \frac{k}{10}$ and $|A'| \leq k$). The probability that none of the αk coordinates are in A is at most $\left(\frac{9}{10}\right)^{\alpha k}$, so

$$\Pr_{\tau,w,(v,J)}[E_1] \leq \Pr_{\tau,w,(v,J)}[E_2] \leq \left(\frac{9}{10}\right)^{\alpha k}. \tag{8}$$

◀

3.3 Local Averaging Operator

In this section we prove the two lemmas concerning local averaging operator. We repeat the two lemmas and prove them.

▶ **Lemma 27.** *For every excellent τ,*

$$\Pr_{w \in [N]^{\frac{9k}{10}}}\left[h_\tau(w) \overset{12\alpha k}{\not\approx} \mathcal{P}_{\frac{3}{4}} h_\tau(w) \;\middle|\; w \in \mathcal{V}_\tau\right] \leq \epsilon^3.$$

Proof. Fix an excellent restriction τ, denote by $\mathcal{V} = \mathcal{V}_\tau$, $h = h_\tau$, $\mathcal{P}_{\frac{3}{4}} h = \mathcal{P}_{\frac{3}{4}} h_\tau$ and $k' = \frac{9k}{10}$. In order to simplify the notations, denote by $\mu = \left(\frac{9}{10}\right)^{\frac{1}{2}\alpha k}$ the constant from the definition of excellence (Definition 23).

From the fact that τ is excellent, we know that $\Pr_{w\in[N]^{k'}}[w \in \mathcal{V}] \geq \frac{\epsilon}{2}$ and

$$\Pr_{w,(v,J)\in\mathcal{N}_{\frac{3}{4}}(w)}\left[w, v \in \mathcal{V}, h_J(w) \overset{\alpha k}{\not\approx} h_J(v)\right] \leq \mu.$$

Our goal is to prove that for almost all $w \in \mathcal{V}$, $\mathcal{P}_{\frac{3}{4}} h(w) \approx h(w)$. First, we characterize the "bad" inputs $w \in \mathcal{V}$ for which we can't prove this claim . Then, we prove it on the rest. Fix $\eta = \epsilon^{20}$, the first set of "bad" inputs is the set of inconsistent ones,

$$B = \left\{ w \in \mathcal{V} \;\middle|\; \Pr_{(v,J)\in\mathcal{N}_{\frac{3}{4}}(w)}\left[v \in \mathcal{V}, h(v)_J \overset{\alpha k}{\not\approx} h(w)_J\right] \geq \frac{\eta}{100} \right\}.$$

By averaging, $\Pr_w[w \in B] \leq \frac{100\mu}{\eta}$.

The second set is the set of "lonely" inputs, inputs that have very sparse neighborhood,

$$L = \left\{ w \in \mathcal{V} \;\middle|\; \Pr_{(v,J)\in\mathcal{N}_{\frac{3}{4}}(w)}[v \in \mathcal{V}] \leq \eta \right\}.$$

By hypercontractivity, Claim 16 (uses [11]), $\Pr_w[w \in L] \leq \eta^{\frac{2}{9}} \left(\frac{\epsilon}{2}\right)^{-\frac{11}{9}}$.

Fix an input $w \in \mathcal{V} \setminus \{B \cup L\}$, we will show that $h(w) \overset{12\alpha k}{\approx} \mathcal{P}_{\frac{3}{4}} h(w)$, i.e. $h(w)$ and $\mathcal{P}_{\frac{3}{4}} h(w)$ are equal on all but $12\alpha k$ of the coordinates. Since $\Pr_w[w \notin B \cap L] \leq \frac{100\mu}{\eta}\eta^{\frac{2}{9}} \left(\frac{\epsilon}{2}\right)^{-\frac{11}{9}} \leq \epsilon^3$, this finishes the proof (ϵ is such that $\epsilon^{120} > \mu$).

Denote by D the following set

$$D = \left\{ i \in [k'] \;\middle|\; h(w)_i \neq \mathcal{P}_{\frac{3}{4}} h(w)_i \right\}.$$

D is the set of coordinates in which the local averaging of h doesn't equal h. Since $w \notin B \cup L$, the neighborhood of w is very consistent, and we show that the set D is small.

Assume towards a contradiction that $|D| > 12\alpha k$. For $v \in [N]^{k'}, J \subset [K]$ and $i \in [k]$, let $E(v, J, i)$ be the event

$$E(v, J, i) = (i \in J \wedge h(w)_i \neq h(v)_i).$$

We will reach a contradiction by upper bounding and lower bounding the probability of the event E, under the distribution $i \in D$ and $(v, J) \in \mathcal{N}_{\frac{3}{4}}(w)$, given that $v \in \mathcal{V}$

Lower bound

We look on $E = E_1 \wedge E_2$, where $E_1 = i \in J$ and $E_2 = h(w)_i \neq h(v)_i$. By definition, for every $i \in D$, the value $h(w)_i$ is not the most probable $h(v)_i$ when $(v, J) \in \mathcal{N}_{\frac{3}{4}}(w)$. Therefore,

$$\forall i \in D, \Pr_{(v,J)\in\mathcal{N}_{\frac{3}{4}}(w)}[E_2 \mid E_1, v \in \mathcal{V}] = \Pr_{(v,J)\in\mathcal{N}_{\frac{3}{4}}(w)}[h(w)_i \neq h(v)_i \mid i \in J, v \in \mathcal{V}] \geq \frac{1}{2}. \quad (9)$$

We want to remove the conditioning over E_1, in order to get a bound E. If we choose a uniform $(v, J) \in \mathcal{N}_{\frac{3}{4}}(w)$, the probability of $i \in J$ is exactly $\frac{3}{4}$. If we condition on $v \in \mathcal{V}$, this probability can be different. We start by bounding the probability of $D \cap J$ to be small.

Every $i \in D$ has probability of $\frac{3}{4}$ be be in J independently, by Chernoff bound (Corollary 10), $\Pr_{(v,J)\in\mathcal{N}_{\frac{3}{4}}(w)}\left[|D\cap J| \leq \frac{3}{5}|D|\right] \leq e^{-\frac{\alpha k}{10}}$. If we condition on $v \in \mathcal{V}$, this probability can increase by a factor of at most $\frac{1}{\eta}$, where $\eta \leq \Pr_{(v,J)\in\mathcal{N}_{\frac{3}{4}}(w)}[v \in \mathcal{V}]$.

$$\Pr_{(v,J)\in\mathcal{N}_{\frac{3}{4}}(w)}\left[|D\cap J| \leq \frac{3}{5}|D| \;\middle|\; v \in \mathcal{V}\right] \leq \frac{1}{\eta}e^{-\frac{\alpha k}{10}}. \tag{10}$$

Equation (10) implies that for a typical $i \in D$, the probability E_1 is not very far from $\frac{3}{4}$. If (v, J) are such that $|D \cap J| \geq \frac{3}{5}|D|$, a random $i \in D$ has probability at least $\frac{3}{5}$ to be in J.

$$\begin{aligned}\Pr_{(v,J)\in\mathcal{N}_{\frac{3}{4}}(w),i\in D}[E_1 \mid v \in \mathcal{V}] &= \Pr_{(v,J)\in\mathcal{N}_{\frac{3}{4}}(w),i\in D}[i \in J \mid v \in \mathcal{V}] \\ &\geq \Pr_{(v,J)\in\mathcal{N}_{\frac{3}{4}}(w),i\in D}\left[i \in J \wedge |D\cap J| \geq \frac{3}{5}|D| \;\middle|\; v \in \mathcal{V}\right] \\ &\qquad \text{(by (10))} \\ &\geq \frac{3}{5}\left(1 - \frac{1}{\eta}e^{-\frac{\alpha k}{10}}\right). \qquad (11)\end{aligned}$$

Now we can lower bound the probability of E:

$$\begin{aligned}\Pr_{(v,J)\in\mathcal{N}_{\frac{3}{4}}(w),i\in D}[E \mid v \in \mathcal{V}] &= \Pr_{(v,J)\in\mathcal{N}_{\frac{3}{4}}(w),i\in D}[E_1 \wedge E_2 \mid v \in \mathcal{V}] \\ &= \Pr_{(v,J)\in\mathcal{N}_{\frac{3}{4}}(w),i\in D}[E_1 \mid v \in \mathcal{V}] \Pr_{(v,J)\in\mathcal{N}_{\frac{3}{4}}(w)}[E_2 \mid E_1, v \in \mathcal{V}] \\ &\qquad \text{(by (9))} \\ &\geq \Pr_{(v,J)\in\mathcal{N}_{\frac{3}{4}}(w),i\in D}[E_1 \mid v \in \mathcal{V}]\frac{1}{2} \qquad \text{(by (11))} \\ &\geq \frac{3}{5}\left(1 - \frac{1}{\eta}e^{-\frac{\alpha k}{10}}\right)\frac{1}{2} \geq \frac{1}{5}. \qquad (12)\end{aligned}$$

Where the last inequality holds since $\eta = \epsilon^{20}$ and ϵ satisfies $\epsilon^{120} > e^{-\frac{\alpha k}{10}}$.

Upper Bound

We want to upper bound the same probability, and reach a contradiction. Since $w \notin L$, $\Pr_{(v,J)\in\mathcal{N}_{\frac{3}{4}}(w)}[v \in \mathcal{V}] \geq \eta$, and from the fact that $w \notin B$ we know that its neighborhood is consistent, i.e. $\Pr_{(v,J)\in\mathcal{N}_{\frac{3}{4}}(w)}\left[v \in \mathcal{V}, h(v)_J \overset{\alpha k}{\not\approx} h(w)_J\right] \leq \frac{\eta}{100}$. Combining both together,

$$\Pr_{(v,J)\in\mathcal{N}_{\frac{3}{4}}(w)}\left[h(v)_J \overset{\alpha k}{\not\approx} h(w)_J \;\middle|\; v \in \mathcal{V}\right] \leq \frac{1}{100}. \tag{13}$$

This implies that with probability at most $\frac{1}{100}$ the chosen (v, J) can be such that $h(v)_J \overset{\alpha k}{\not\approx} h(w)_J$.

Else, $h(v)_J \overset{\alpha k}{\approx} h(w)_J$, so there are at most αk coordinates $i \in J$ in which $h(v)_i \neq h(w)_i$. Since $|D| \geq 12\alpha k$, with probability at most $\frac{1}{12}$ a uniform $i \in D$ is in these αk coordinates.

$$\Pr_{(v,J)\in\mathcal{N}_{\frac{3}{4}}(w),i\in D}[E \mid v \in \mathcal{V}] \leq \frac{1}{100} + \frac{1}{12} < \frac{1}{5}. \tag{14}$$

And we reached a contradiction with (12). ◀

In order to show that the function $\mathcal{P}_{\frac{3}{4}}h_\tau$ is close to a product function, we need to show that it is consistent in a similar way to h_τ (as in the definition of excellence, Definition 23). Lemma 27 only gives us that $\mathcal{P}_{\frac{3}{4}}h_\tau$ is consistent among the inputs in $\mathcal{V}_\tau$, and not in all $[N]^{\frac{9k}{10}}$.

▶ **Lemma 28.** *For every excellent τ,*

$$\Pr_{w,(v,J)}\left[\mathcal{P}_{\frac{3}{4}}h(w)_J \overset{20\alpha k}{\approx} \mathcal{P}_{\frac{3}{4}}h(v)_J\right] \geq 1-\epsilon^{10},$$

where $w \in [N]^{\frac{9k}{10}}$ and $(v,J) \in \mathcal{N}_{\frac{1}{2}}(w)$.

Proof. This proof is similar to the proof of Lemma 27. We fix excellent τ and denote $\mathcal{V}=\mathcal{V}_\tau$, $h=h_\tau$ and $\mathcal{P}_{\frac{3}{4}}h=\mathcal{P}_{\frac{3}{4}}h_\tau$, $k'=\frac{9k}{10}$ and $\mu=\left(\frac{9}{10}\right)^{\frac{1}{2}\alpha k}$.

We characterize the inputs $w,(v,J)$ on which we can't prove that $\mathcal{P}_{\frac{3}{4}}h(w)_J \overset{20\alpha k}{\approx} \mathcal{P}_{\frac{3}{4}}h(v)_J$. Instead of the set B in the proof of Lemma 27, we define a set of two correlated inputs $(w,(v,J))$ that are inconsistent. Fixing $\eta=\epsilon^{51}$, let

$$C=\left\{w,(v,J)\;\middle|\; \Pr_{(w',J'),(v',J'')}\left[w',v'\in\mathcal{V}, h(w')_{\tilde{J}} \overset{\alpha k}{\not\approx} h(v')_{\tilde{J}}\right] \geq \frac{\eta^2}{4000}\right\}.$$

Where $(w',J')\in\mathcal{N}_{\frac{3}{4}}(w),(v',J'')\in\mathcal{N}_{\frac{3}{4}}(v)$ and $\tilde{J}=J\cap J'\cap J''$.

If w is chosen uniformly in $[N]^{k'}$ and $(v,J)\in\mathcal{N}_{\frac{1}{2}}(w)$, then the marginal distribution on w' is uniform, and $(v',\tilde{J})\in\mathcal{N}_{\left(\frac{3}{4}\right)^2\frac{1}{2}}(w')$, since for each i independently, the probability of i to be in $\tilde{J}=J\cap J'\cap J''$ is $\left(\frac{3}{4}\right)^2\frac{1}{2}$.

Since τ is excellent, $\Pr_{w',(v',\tilde{J})}\left[w',v'\in\mathcal{V}, h(w')_{\tilde{J}} \overset{\alpha k}{\not\approx} h(v')_{\tilde{J}}\right] \leq \mu$. By averaging, it means that $\Pr_{w,(v,J)}[w,(v,J)\in C]\leq\frac{4000\mu}{\eta^2}$.

We define the set of inputs with sparse neighborhood,

$$L=\left\{w\in[N]^{k'}\;\middle|\; \Pr_{(w',J')\in\mathcal{N}_{\frac{3}{4}}(w)}[w'\in\mathcal{V}]\leq\eta\right\}.$$

From hypercontractivity argument, see Claim 16, $\Pr_w[w\in L]\leq\eta^{\frac{2}{9}}\left(\frac{\epsilon}{2}\right)^{-\frac{11}{9}}$.

For every w and (v,J) such that $w,v\notin L$ and $(w,(v,J))\notin C$, we show that $\mathcal{P}_{\frac{3}{4}}h(w)_J \overset{20\alpha k}{\not\approx} \mathcal{P}_{\frac{3}{4}}h(v)_J$. This finishes the proof since for $w\in[N]^{k'}$ and $(v,J)\in\mathcal{N}_{\frac{1}{2}}(w)$,

$$\Pr_{w,(v,J)}[(w,(v,J))\in C\vee w\in L\vee v\in L]\leq\frac{4000\mu}{\eta^2}+2\cdot\eta^{\frac{2}{9}}\left(\frac{\epsilon}{2}\right)^{-\frac{11}{9}}\leq\epsilon^{10}.$$

Fix $w,(v,J)$ such that $w,v\notin L$ and $(w,(v,J))\notin C$, and let $D\subseteq J$ be the set

$$D=\left\{i\in J\;\middle|\; \mathcal{P}_{\frac{3}{4}}h(w)_i\neq\mathcal{P}_{\frac{3}{4}}h(v)_i\right\}.$$

Similarly to the previous proof, we assume towards a contradiction that $|D|\geq 20\alpha k$.

For every $J',J''\subset[k'],w',v'\in\mathcal{V}$ and $i\in[k']$, we denote by $E(J',J'',w',v',i)$ the following event:

$$E(J',J'',w',v',i)=\left(h(w')_i\neq h(v')_i\wedge i\in J'\cap J''\right).$$

We upper bound and lower bound the probability of this event, under the distribution $i\in D$ and $(w',J')\in\mathcal{N}_{\frac{3}{4}}(w),(v',J'')\in\mathcal{N}_{\frac{3}{4}}(v)$ given that $w',v'\in\mathcal{V}$.

Lower Bound

We look on $E = E_1 \wedge E_2$, where $E_1 = i \in J' \cap J''$ and $E_2 = h(w')_i \neq h(v')_i$.

For every $i \in D$, $\mathcal{P}_{\frac{3}{4}} h(w)_i \neq \mathcal{P}_{\frac{3}{4}} h(v)_i$, so the most frequent value $h(w')_i$ for $(w', J') \in \mathcal{N}_{\frac{3}{4}}(w)$ doesn't equal the most frequent value $h(v')_i$ for $(v', J'') \in \mathcal{N}_{\frac{3}{4}}(v)$. For every $i \in D$, taking $(w', J') \in \mathcal{N}_{\frac{3}{4}}(w), (v', J'') \in \mathcal{N}_{\frac{3}{4}}(v)$:

$$\Pr_{\substack{(w',J')\\(v',J'')}}[E_2 \mid E_1, w', v' \in \mathcal{V}] = \Pr_{\substack{(w',J')\\(v',J'')}}[h(w')_i \neq h(v')_i \mid i \in J' \cap J'', w', v' \in \mathcal{V}] \geq \frac{1}{2}. \tag{15}$$

In order to prove the lower bound, we need to remove the condition over E_1. To do that, we need to lower bound the size of $D \cap J' \cap J''$. Both J' and J'' are taken by picking each coordinated independently with probability $\frac{3}{4}$. If we do not condition on $w', v' \in \mathcal{V}$ expected value of $|D \cap J' \cap J''|$ is $\left(\frac{3}{4}\right)^2 |D|$. Each $i \in D$ is in $J' \cap J''$ with probability $\left(\frac{3}{4}\right)^2$ independently, so using Chernoff bound (Corollary 10),

$$\Pr_{\substack{(w',J') \in \mathcal{N}_{\frac{3}{4}}(w)\\(v',J'') \in \mathcal{N}_{\frac{3}{4}}(v)}}[|D \cap J' \cap J''| \leq 0.56|D|] \leq e^{-\frac{\alpha k}{9000}}.$$

If we condition on $w' \in \mathcal{V}, v' \in \mathcal{V}$, the probability can increase by a factor of at most $\frac{1}{\eta^2}$, where $\Pr_{(w',J') \in \mathcal{N}_{\frac{3}{4}}(w)}[w' \in \mathcal{V}] \geq \eta$ and $\Pr_{(v',J'') \in \mathcal{N}_{\frac{3}{4}}(v)}[v' \in \mathcal{V}] \geq \eta$,

$$\Pr_{\substack{(w',J') \in \mathcal{N}_{\frac{3}{4}}(w)\\(v',J'') \in \mathcal{N}_{\frac{3}{4}}(v)}}[|D \cap J' \cap J''| \leq 0.56|D| \mid w', v' \in \mathcal{V}] \leq \frac{1}{\eta^2} e^{-\frac{\alpha k}{9000}}. \tag{16}$$

If $|D \cap J' \cap J''| \geq 0.56|D|$, then a uniform $i \in D$ has probability of at least 0.56 to be in $J' \cap J''$,

$$\Pr_{\substack{i \in D, (w',J') \in \mathcal{N}_{\frac{3}{4}}(w)\\(v',J'') \in \mathcal{N}_{\frac{3}{4}}(v)}}[E_1 \mid w', v' \in \mathcal{V}] \geq \left(1 - \frac{1}{\eta^2} e^{-\frac{\alpha k}{9000}}\right) 0.56 \geq 0.55. \tag{17}$$

The last inequality is correct because we assume ϵ is large enough to satisfy $\epsilon^{120} > \frac{1}{\eta^2} e^{-\frac{\alpha k}{9000}}$.

Combining (15) and (17), we can lower bound the probability of E, when $i \in D, (w', J') \in \mathcal{N}_{\frac{3}{4}}(w)$ and $(v', J'') \in \mathcal{N}_{\frac{3}{4}}(v)$,

$$\begin{aligned}
\Pr_{i,(w',J'),(v',J'')}[E \mid w', v' \in \mathcal{V}] &= \Pr_{i,(w',J'),(v',J'')}[E_1 \wedge E_2, \mid w', v' \in \mathcal{V}] \\
&= \Pr_{i,(w',J'),(v',J'')}[E_1 \mid w', v' \in \mathcal{V}] \qquad (18)\\
&\quad \cdot \Pr_{i,(w',J'),(v',J'')}[E_2 \mid w', v' \in \mathcal{V}, E_1] \\
&\geq \frac{1}{2} \cdot 0.55 > \frac{1}{4}. \qquad (19)
\end{aligned}$$

Upper Bound

Since $(w, (v, J)) \notin C$, we know that

$$\Pr_{\substack{(w',J') \in \mathcal{N}_{\frac{3}{4}}(w)\\(v',J'') \in \mathcal{N}_{\frac{3}{4}}(v)}}\left[w', v' \in \mathcal{V}, h(w')_{\bar{J}} \overset{\alpha k}{\not\approx} h(v')_{\bar{J}}\right] \leq \frac{\eta^2}{4000},$$

where $\tilde{J} = J \cap J' \cap J''$. From the fact that $w \notin L$, $\Pr_{(w',J')\in\mathcal{N}_{\frac{3}{4}}(w)}[w' \in \mathcal{V}] \geq \eta$ and since $v \notin L$, $\Pr_{(v',J'')\in\mathcal{N}_{\frac{3}{4}}(v)}[v' \in \mathcal{V}] \geq \eta$. This implies that

$$\Pr_{\substack{(w',J')\in\mathcal{N}_{\frac{3}{4}}(w)\\(v',J'')\in\mathcal{N}_{\frac{3}{4}}(v)}}\left[h(w')_{\tilde{J}} \overset{\alpha k}{\not\approx} h(v')_{\tilde{J}} \;\middle|\; w', v' \in \mathcal{V}\right] \leq \frac{1}{4000}.$$

If $h(w')_{\tilde{J}} \overset{\alpha k}{\approx} h(v')_{\tilde{J}}$, then even if all these αk coordinates are in D, a uniform $i \in D$ has probability of at most $\frac{\alpha k}{|D|} \leq \frac{\alpha k}{20\alpha k} \leq \frac{1}{20}$ to be one of these coordinates. Therefore,

$$\Pr_{\substack{i\in D,(w',J')\in\mathcal{N}_{\frac{3}{4}}(w)\\(v',J'')\in\mathcal{N}_{\frac{3}{4}}(v)}}[E] \leq \frac{1}{4000} + \frac{1}{20} < \frac{1}{10},$$

which contradicts (19). ◀

3.4 Direct Product Function

Fixing an excellent τ, we first show that the local average function $\mathcal{P}_{\frac{3}{4}}h_\tau$ is close to a product function g^τ. Then, by Lemma 27, we will conclude that h_τ is close to g^τ. This implies that τ is a DP restriction as needed.

In this section we prove Lemma 29,

▶ **Lemma 29.** *For every excellent restriction τ there exist a product function $g^\tau : [N]^{\frac{9k}{10}} \to [M]^{\frac{9k}{10}}$ such that*

$$\Pr_{w\in[N]^{\frac{9k}{10}}}\left[\mathcal{P}_{\frac{3}{4}}h_\tau(w) \overset{1500\alpha k}{\not\approx} g^\tau(w)\right] \leq 3\epsilon^4.$$

We first define g^τ, the candidate direct product function

▶ **Definition 30.** For each excellent $\tau = (A, r, \gamma)$, let $g^\tau : [N]^{\frac{9k}{10}} \to [M]^{\frac{9k}{10}}$ be the following function, for each $i \notin A$ and $b \in [N]$,

$$g_{\tau,i}(b) = \underset{w\in[N]^{\frac{9k}{10}} \text{ s.t. } w_i=b}{\text{Plurality}} \{\mathcal{P}_{\frac{3}{4}}h(w)_i\},$$

ties are broken arbitrarily.

We prove Lemma 29 using the following few claims.

▶ **Claim 31.**

$$\Pr_{i\in\left[\frac{9k}{10}\right],w,v\in[N]^{\frac{9k}{10}}}\left[\mathcal{P}_{\frac{3}{4}}h(w)_i = \mathcal{P}_{\frac{3}{4}}h(v)_i \;\middle|\; w_i = v_i\right] \geq 1 - 200\alpha.$$

In order to prove Claim 31, we need to define an "almost ρ-correlated"' distribution.

▶ **Definition 32.** (x, J) are almost ρ-correlated to $y \in [N]^k$, denoted by $(x, J) \in A_\rho(y)$, if they are chosen by the following process:

1. Choose $i \in [k]$ uniformly at random, set $J = \{i\}$.
2. For each $j \neq i$, add j to J with probability ρ independently.
3. Set $x_J = y_J$ and the rest of x is uniform.

▶ **Claim 33.** *For any $\rho \in (0,1)$ and any event $E(y,x,J)$ over $x,y \in [N]^k$ and $J \subseteq [k]$,*

$$\Pr_{y\in[N]^k,(x,J)\in A_\rho(y)}[E(y,x,J)] \leq 2 \Pr_{y\in[N]^k,(x,J)\in \mathcal{N}_\rho(y)}[E(y,x,J)] + 5e^{-\frac{\rho k}{4}}.$$

The proof appears at the end of the section.

Proof of Claim 31. Let $k' = \frac{9k}{10}$. We start by showing that for a uniform $w \in [N]^{k'}$, $(u,J') \in A_{\frac{1}{2}}(w)$ and $i \in J'$, $\Pr_{i,w,(v,J')}\left[\mathcal{P}_{\frac{3}{4}}h(w)_i = \mathcal{P}_{\frac{3}{4}}h(v)_i\right] \geq 1 - 100\alpha$.

Let E_1 be the event that $\mathcal{P}_{\frac{3}{4}}h(w)_i \neq \mathcal{P}_{\frac{3}{4}}h(v)_i$, we further define the following two events, let E_2 to be the event $\mathcal{P}_{\frac{3}{4}}h(w)_{J'} \overset{20\alpha k}{\not\approx} \mathcal{P}_{\frac{3}{4}}h(u)_{J'}$, and let E_3 be the event that $|J'| < \frac{k}{4}$.

If both E_2, E_3 don't happen, then $|J'| \geq \frac{k}{4}$, and there are at most $20\alpha k$ coordinates i in which $\mathcal{P}_{\frac{3}{4}}h(w)_i \neq \mathcal{P}_{\frac{3}{4}}h(v)_i$. Therefore, a uniform $i \in J'$ has probability at most $\frac{20\alpha k}{\frac{k}{4}}$ to satisfy $\mathcal{P}_{\frac{3}{4}}h(w)_i \neq \mathcal{P}_{\frac{3}{4}}h(v)_i$,

$$\Pr_{w,(v,J'),i}[E_1 \mid \neg E_2, \neg E_3] \leq \frac{20\alpha k}{\frac{k}{4}} = 80\alpha. \tag{20}$$

In order to remove the condition over $\neg E_2, \neg E_3$, we bound their probability. For a uniform $w \in [N]^{k'}$ and $(u,J') \in A_{\frac{1}{2}}(\rho)$,

$$\begin{aligned}\Pr_{w,(u,J')\in A_{\frac{1}{2}}(w)}[E_2] &\leq 2 \Pr_{w,(u,J')\in \mathcal{N}_{\frac{1}{2}}(w)}[E_2] + 5e^{-\frac{\rho k}{4}} && \text{(by Claim 33)}\\ &\leq 2\epsilon^{10} + 5e^{-\frac{\rho k}{4}} \leq 3\epsilon^{10}. && \text{(by Lemma 28)}\end{aligned}$$

Similarly, for $w \in [N]^{k'}$ and $(u,J') \in A_{\frac{1}{2}}(w)$,

$$\begin{aligned}\Pr_{w,(u,J')\in A_{\frac{1}{2}}(w)}[E_3] &\leq 2 \Pr_{w,(u,J')\in \mathcal{N}_{\frac{1}{2}}(w)}[E_3] + 5e^{-\frac{\rho k}{4}} && \text{(by Claim 33)}\\ &\leq 2e^{-\frac{k}{100}} + 5e^{-\frac{\rho k}{4}} \leq \epsilon^{10}. && \text{(Chernoff Bound)}\end{aligned}$$

For $(u,J') \in \mathcal{N}_{\frac{1}{2}}(w)$, each coordinate i is in J' with probability $\frac{1}{2}$ independently, so we can use Chernoff bound. If we add a condition on $\neg E_2$, it can increase the probability by a factor of $\frac{1}{\Pr[\neg E_2]} < 2$, therefore $\Pr_{w,(u,J')\in A_{\frac{1}{2}}(w)}[E_3 \mid \neg E_2] \leq 2\epsilon^{10}$.

Combining everything together, for a uniform $w \in [N]^{k'}$, $(u,J') \in A_{\frac{1}{2}}(w)$ and $i \in J'$,

$$\begin{aligned}\Pr_{w,(u,J'),i}[E_1] &\leq \Pr_{w,(u,J'),i}[E_2] + \Pr_{w,(u,J'),i}[E_3 \mid \neg E_2] + \Pr_{w,(u,J'),i}[E_1 \mid \neg E_2, \neg E_3]\\ &\leq 3\epsilon^{10} + 2\epsilon^{10} + 80\alpha \leq 100\alpha \end{aligned}\tag{21}$$

Let $\mathcal{D}' : [k'] \times [N]^{k'} \times [N]^{k'} \times [N]^{k'} \to \{0,1\}$ be the following distribution, generating i, w, v, u:

1. Pick a uniform $i \in [k']$.
2. Pick $w, v \in [N]^{k'}$ such that $w_i = v_i$.
3. For every $j \neq i$, insert j into J with probability $\frac{1}{2}$ independently.
4. For every $j \in [k']$, $u_j = \begin{cases} w_j & j \in J \\ v_j & \text{else} \end{cases}$.

The distribution $\mathcal{D}'$ is built such that the marginal distribution over w, v, i is that $i \in [k']$ uniformly, and w, v are uniform in $[N]^{k'}$ such that $w_i = v_i$. Furthermore, the marginal distribution over $w, (u, J \cup \{i\}), i$ is such that $w \in [N]^{k'}$ uniformly, $(u, J \cap \{i\}) \in A_{\frac{1}{2}}(w)$ and the coordinate i is uniform in $\{i\} \cup J$. Similarly, the marginal distribution over $v, (u, \bar{J})$ is $v \in [N]^{k'}$, $(u, \bar{J}) \in A_{\frac{1}{2}}(v)$ and $i \in \bar{J}$.

Therefore, we can use equation (21) on the pairs $w, (u, J \cup \{i\})$ and $v, (u, \bar{J})$, and by union bound,

$$\Pr_{\substack{i\in[k']\\ w,v\in[N]^{k'}}}\left[\mathcal{P}_{\frac{3}{4}}h(w)_i = \mathcal{P}_{\frac{3}{4}}h(v)_i \;\middle|\; w_i = v_i\right] \geq \Pr_{i,w,v,u\sim\mathcal{D}'}\left[\mathcal{P}_{\frac{3}{4}}h(w)_i = \mathcal{P}_{\frac{3}{4}}h(v)_i = \mathcal{P}_{\frac{3}{4}}h(u)_i\right]$$
$$\geq 1 - 100\alpha - 100\alpha.$$

◀

▶ **Corollary 34.**

$$\Pr_{w\in[N]^{\frac{9k}{10}},i\in[\frac{9k}{10}]}\left[\mathcal{P}_{\frac{3}{4}}h(w)_i = g(w)_i\right] \geq 1 - 400\alpha.$$

Proof. For each $w \in [N]^{\frac{9k}{10}}$ and $i \in [\frac{9k}{10}]$ such that $\mathcal{P}_{\frac{3}{4}}h(w)_i \neq g(w)_i$, the value $\mathcal{P}_{\frac{3}{4}}h(w)_i$ is not the most frequent, $\Pr_{v\in[N]^{\frac{9k}{10}}}[\mathcal{P}_{\frac{3}{4}}h(w)_i = \mathcal{P}_{\frac{3}{4}}h(v)_i | w_i = v_i] \leq \frac{1}{2}$. Therefore,

$$\Pr_{w,v\in[N]^{\frac{9k}{10}},i\in[\frac{9k}{10}]}\left[\mathcal{P}_{\frac{3}{4}}h(w)_i \neq \mathcal{P}_{\frac{3}{4}}h(v)_i \;\middle|\; w_i = v_i\right] \geq \frac{1}{2}\Pr_{w\in[N]^{\frac{9k}{10}},i\in[\frac{9k}{10}]}\left[\mathcal{P}_{\frac{3}{4}}h(w)_i \neq g(w)_i\right].$$

Using Claim 31 we reach the corollary. ◀

Proof of Lemma 29. Fix an excellent τ, denote $k' = \frac{9k}{10}$.

For each $w \in [N]^{k'}$, let $D_w \subset [k']$ be the set of coordinates in which $g^\tau(w), \mathcal{P}_{\frac{3}{4}}h_\tau(w)$ differs

$$D_w = \left\{ i \in [k'] \;\middle|\; g^\tau(w)_i \neq \mathcal{P}_{\frac{3}{4}}h_\tau(w)_i \right\}.$$

Let $C \subset [N]^{k'}$ be the set of inputs such that $g^\tau, \mathcal{P}_{\frac{3}{4}}h_\tau$ are similar on them,

$$C = \left\{ w \in [N]^{k'} \;\middle|\; |D_w| \leq 500\alpha k \right\}.$$

By Corollary 34 and averaging, $\Pr_w[w \in C] \geq \frac{1}{5}$.

Let $B \subset [N]^{k'}$ be the set of inputs on which $g^\tau, \mathcal{P}_{\frac{3}{4}}h_\tau$ are far,

$$B = \left\{ w \in [N]^{k'} \;\middle|\; |D_w| \geq 1500\alpha k \right\}.$$

B is the set of inputs in which $\mathcal{P}_{\frac{3}{4}}h_\tau(w) \overset{1500\alpha k}{\not\approx} g^\tau(w)$, so our goal is to prove that B is small.

Let $E_1(w, v, J)$ be the event that $|J \cap D_w| > 600\alpha k$, and let $E_2(w, v, J)$ be the event that $\mathcal{P}_{\frac{3}{4}}h(w)_J \overset{20\alpha}{\not\approx} \mathcal{P}_{\frac{3}{4}}h(v)_J$. By Lemma 28, $\Pr_{w\in[N]^{k'},(v,J)\in\mathcal{N}_{\frac{1}{2}}(w)}[E_2] \leq \epsilon^{10}$.

For every v, w, J such that $v_J = w_J$, the function g satisfies $g^\tau(w)_J = g^\tau(v)_J$ (since g is a product function), and therefore $E_1 \wedge (v \in C) \implies E_2$. This is because if E_2 doesn't hold, then $\mathcal{P}_{\frac{3}{4}}h(w)_J \overset{20\alpha}{\approx} \mathcal{P}_{\frac{3}{4}}h(v)_J$, if E_1 does hold then $|J \cap D_w| > 600\alpha k$, which means that $|D_v \cap J| \geq 580\alpha k$, and $v \notin C$.

We show if B isn't small, then $E_1 \wedge (v \in C)$ happens often, when we pick $w \in [N]^k, (v,J) \in \mathcal{N}_{\frac{1}{2}}(w)$.

For $w \in B$, the set D_w is large, $|D_w| \geq 1500\alpha k$, if we take $(v,J) \in \mathcal{N}_{\frac{1}{2}}(w)$, each coordinate $i \in D_w$ is in J with probability $\frac{1}{2}$ independently, so for $w \in B$, by Chernoff bound

$$\Pr_{(v,J)\in\mathcal{N}_{\frac{1}{2}}(w)}[E_1(w)] = \Pr_{(v,J)\in\mathcal{N}_{\frac{1}{2}}(w)}[|J \cap D_w| > 600\alpha k] \geq 1 - e^{-\frac{\rho k}{100}}.$$

From reverse hypercontractivity [11], Corollary 15

$$\Pr_{w,(v,J)\in\mathcal{N}_{\frac{1}{2}}(w)}[w \in B, v \in C] \geq \Pr_w[w \in C]^{\frac{3}{2}} \Pr_w[w \in B]^{\frac{5}{2}}.$$

Therefore,

$$\begin{aligned}\Pr_{w,(v,J)\in\mathcal{N}_{\frac{1}{2}}(w)}[w \in B, v \in C \wedge E_1] &\geq \Pr_w[w \in C]^{\frac{3}{2}} \Pr_w[w \in B]^{\frac{5}{2}} - e^{-\frac{\rho k}{100}} \\ &\geq \left(\frac{1}{5}\right)^{\frac{3}{2}} \Pr_w[w \in B]^{\frac{5}{2}} - e^{-\frac{\rho k}{100}}. \qquad (22)\end{aligned}$$

Where (22) is since $\Pr_w[w \in C] \geq \frac{1}{5}$.

Since $E_1 \wedge (v \in C) \implies E_2$ and by Lemma 28, $\Pr_{w\in[N]^{k'},(v,J)\in\mathcal{N}_{\frac{1}{2}}(w)}[E_2] \leq \epsilon^{10}$, it means that (22) should be smaller than ϵ^{10}, which implies $\Pr_w[w \in B] \leq 3\epsilon^4$ and finishes the proof. ◀

We are left with proving the simple distribution claim – that almost ρ correlated is similar to ρ correlated.

Proof of Claim 33. The proof is based on the fact that the distributions $\mathcal{N}_\rho(y), A_\rho(y)$ are very close, and the probability of an event depending on y, x, J is not much different in both distributions.

By Chernoff bound, ρ-correlated sets are almost always of size about ρk, this holds for almost ρ correlated as well,

$$\Pr_{(x,J)\in\mathcal{N}_\rho(y)}[|J| > 2\rho k] \leq e^{-\frac{\rho k}{3}},$$

$$\Pr_{(x,J)\in A_\rho(y)}[|J| > 2\rho k] \leq e^{-\frac{\rho k}{4}}.$$

For each $y \in [N]^k$, let B_y be the (x,J) that satisfy $E(y,x,J)$

$$B_y = \{(x,J) \mid E(y,x,J) = 1\}.$$

Using this notation

$$\Pr_{y\in[N]^k,(x,J)\in\mathcal{N}_\rho(y)}[E(y,x,J)] = \Pr_{y\in[N]^k,(x,J)\in\mathcal{N}_\rho(y)}[(x,J) \in B_y].$$

Fix $y \in [N]^k$, for each $(x,J) \in B_y$, by the definition of ρ-correlation,

$$\Pr_{(z,J')\in\mathcal{N}_\rho(y)}[(z,J') = (x,J)] = \rho^{|J|}(1-\rho)^{k-|J|}\left(\frac{1}{N}\right)^{k-|J|}.$$

By the definition of almost ρ correlation,

$$\Pr_{(z,J')\in A_\rho(y)}[(z,J')=(x,J)]=\frac{|J|}{k}\rho^{|J|-1}(1-\rho)^{k-|J|}\left(\frac{1}{N}\right)^{k-|J|}.$$

Note that for each such $(x,J)\in B_y$ such that $|J|\le 2\rho k$,

$$\Pr_{(z,J')\in A_\rho(y)}[(z,J')=(x,J)]\le 2\Pr_{(z,J')\in \mathcal{N}_\rho(y)}[(z,J')=(x,J)].$$

Therefore

$$\begin{aligned}\Pr_{(x,J)\in A_\rho(y)}[(x,J)\in B_y]\le&\Pr_{(x,J)\in A_\rho(y)}[|J|\ge 2\rho k]+\Pr_{(x,J)\in A_\rho(y)}[(x,J)\in B_y\mid |J|\le 2\rho k]\\ \le&e^{-\frac{\rho k}{4}}+2\Pr_{(x,J)\in \mathcal{N}_\rho(y)}[(x,J)\in B_y\mid |J|\le 2\rho k]\\ \le&e^{-\frac{\rho k}{4}}+2\Pr_{(x,J)\in \mathcal{N}_\rho(y)}[(x,J)\in B_y]+4e^{-\frac{\rho k}{3}}.\end{aligned}$$

When we used conditional probability in the last inequality. This is true for all $y\in[N]^{k'}$, therefore,

$$\begin{aligned}\Pr_{y\in[N]^k,(x,J)\in A_\rho(y)}[E(y,x,J)]=&\Pr_{y\in[N]^k,(x,J)\in A_\rho(y)}[(x,J)\in B_y]\\ \le&2\Pr_{y\in[N]^k,(x,J)\in \mathcal{N}_\rho(y)}[(x,J)\in B_y]+5e^{-\frac{\rho k'}{4}}.\end{aligned}$$

◀

4 Global Structure for Sets

Up until now we have considered functions $f:[N]^k\to[M]^k$ whose inputs are ordered tuples $(x_1\ldots,x_k)\in[N]^k$. We now move to consider functions $f:\binom{[N]}{k}\to[M]^k$ whose inputs are unordered $\{x_1,\ldots,x_k\}\in\binom{[N]}{k}$, and we assume that $N\gg k$ (for tuples no such assumption was made).

To each subset $S=\{s_1,\ldots,s_k\}$ the function f assigns $f(S)\in[M]^k$. $f(S)$ should be viewed as a "local function" on S, assigning a value from $[M]$ to every $a\in S$. We denote by $f(S)_a$ the output of f that corresponds to a. For a subset $W\subset S$, let $f(S)_W$ be the outputs of f corresponding to the elements in W.

There are straightforward analogs to Theorem 1 and Theorem 21 which we present and prove in this section. Interestingly, in the case of sets deducing global structure from restricted global structure is quite easier than it is for tuples.

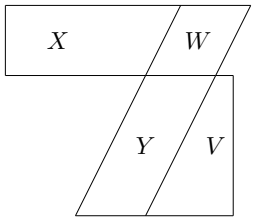
First, let us present the Z test for sets, from [9] when $t=\frac{k}{10}$. Let $\alpha_{Z_{set}(\frac{k}{10})}(f)$ be the success probability of this test. This is the same test as Test 3 from the introduction written differently, it is written this way because it is easier to refer to the test items during the proof.

▶ **Theorem 5.** *There exist a small constant $c>0$, such that for every constant $\lambda>0$, large enough $k\in\mathbb{N}$ and $N>k^2e^{10c\lambda k}$, if the function $f:\binom{[N]}{k}\to[M]^k$ passes Test 3 with probability $\alpha_{Z_{set}(\frac{k}{10})}(f)=\epsilon>e^{-c\lambda k}$, then there exist a function $g:[N]\to[M]$ such that*

$$\Pr_S\left[f(S)\overset{\lambda k}{\approx}g(S)\right]\ge\epsilon-4\epsilon^2.$$

In order to analyze this test, we first need to "translate" the restricted global structure result into this setting, and then prove the global structure in this setting.

1. Choose a random set $W \subset [N]$ of size $\frac{k}{10}$.
2. Choose $X, Y \subset [N] \setminus W$ of size $\frac{9k}{10}$.
3. If $f(X \cup W)_W \neq f(Y \cup W)_W$ reject.
4. Choose $V \subset [N] \setminus Y$ of size $\frac{k}{10}$.
5. If $f(Y \cup W)_Y \neq f(Y \cup V)_Y$ reject, else accept.



Denote by $\alpha_{Z_{set}(\frac{k}{10})}(f)$ the success probability of f on this test.

Test 4 "Z" test for functions over sets, with $t = \frac{k}{10}$ (3-query test).

4.1 Restricted Global Structure for Sets

In this section, we see that for $N \gg k$, the restricted global structure for tuples, Theorem 21, implies restricted global structure for sets. First we define analog definitions for sets, for good restrictions and DP restrictions. To make the reduction proof simpler, we use a constant $\eta \in \left[1 - \frac{k^2}{N}, 1\right]$ (i.e. almost 1) and define good pair using η.

▶ **Definition 35** (Good pair). A pair $X, W \subset [N]$, $|X| = \frac{9k}{10}, |W| = \frac{k}{10}$ is good if

$$\Pr_Y [f(X \cup W)_W = f(Y \cup W)_W \mid Y \cap W = \emptyset] > \frac{\epsilon}{2}\eta.$$

This definition is analog to Definition 19 of good restriction, the main difference between the definitions is that here we don't have a set of coordinates $A \subset [k]$, because f is defined on sets and not coordinates.

▶ **Definition 36** (DP pair). A pair $X, W \subset [N]$, $|X| = \frac{9k}{10}, |W| = \frac{k}{10}$ is a DP pair if it is good, and if there exist a function $g_{X,W} : [N] \to [M]$ such that

$$\Pr_Y \left[f(Y \cup W)_Y \overset{3\alpha k}{\not\approx} g_{X,W}(Y) \;\middle|\; Y \cap W = \emptyset, f(X \cup W)_W = f(Y \cup W)_W \right] \leq 2\epsilon^2.$$

This definition is analog to Definition 20 of DP restriction, only here there is a single function $g_{X,W}$, instead of $\frac{9k}{10}$ different functions in the case of coordinates.

▶ **Lemma 37** (Restricted global structure for sets). *There exist a small constants $\delta > 0$, such that for every constant $\lambda > 0$ and large enough $k \in \mathbb{N}$ such that $N > k^2 e^{10\delta\lambda k}$, the following holds,*

For every function $f : \binom{[N]}{k} \to [M]^k$, if $\alpha_{Z_{set}(\frac{k}{10})}(f) = \epsilon > e^{-\delta\lambda k}$, then at least $(1 - \epsilon^2 - \frac{k^2}{N})$ of the good pairs $W \in \binom{[N]}{\frac{k}{10}}, X \in \binom{[N]}{\frac{9k}{10}}$ are DP pairs, i.e. there exist $g_{X,W} : [N] \to [M]$ such that

$$\Pr_Y \left[f(Y \cup W)_Y \overset{3\alpha k}{\not\approx} g_{X,W}(Y) \;\middle|\; Y \cap W = \emptyset, f(X \cup W)_W = f(Y \cup W)_W \right] \leq 2\epsilon^2.$$

This lemma for sets is analog to Theorem 21, and we prove it by a reduction from it. For every $f : \binom{[N]}{k} \to [M]^k$ we define a function $f' : [N]^k \to [M]^k \cup \perp$ that equals $\perp$ if the input has two identical coordinates, and identifies with f everywhere else. For $N \gg k$, almost all inputs don't have two identical coordinates, and f', f are equal almost always.

Using Theorem 21, we derive a restricted global structure on f' which gives a direct product function $g^\tau = g^\tau_1, \ldots g^\tau_{\frac{9k}{10}k}$ for every excellent τ. Since f equals f' almost always, we find an equivalence between excellent τ and excellent X, W. Then, we build a restricted

global function $g_{X,W}$ by taking the most frequent value among the product $g_1^\tau, \dots g_{\frac{9k}{10}}^\tau$. Note that even though f' is permutation invariant, the functions $g_1^\tau, \dots g_{\frac{9k}{10}}^\tau$ may not be the same.

Since the proof is technical, and its main points are described in the paragraph above, we defer it to Appendix B.

4.2 Global Structure for Sets

Now we are ready to prove Theorem 5. The proof is very similar to lemma 3.16 in [9].

Proof. Fix a function $f : \binom{[N]}{k} \to [M]^k$ that passes Test 4 with probability $\epsilon > e^{-c\lambda k}$, denote by $\delta = \frac{c}{5}$ and $\alpha = 5\lambda$.

Let $W \in \binom{[N]}{\frac{k}{10}}, X \in \binom{[N]}{\frac{9k}{10}}$ be the subsets chosen on the first two items of the test, if $\Pr_Y \left[f(X \cup W)_W = f(Y \cup W)_W \mid Y \cap W = \emptyset\right] < \frac{\epsilon}{2}\eta$, the test rejects in Item 3 with probability at least $1 - \frac{\epsilon}{2}\eta$.

Therefore, in order for f to pass the test with probability ϵ, the test must pass with probability at least ϵ on W, X such that $\Pr_Y \left[f(X \cup W)_W = f(Y \cup W)_W \mid Y \cap W = \emptyset\right] > \frac{\epsilon}{2}\eta$, we call these W, X good.

Using Lemma 37, for at least $(1 - 2\epsilon^2 - \frac{k^2}{N})$ of the good W, X there exist a function $g_{W,X} : [N] \to [M]$ such that

$$\Pr_Y \left[f(Y \cup W)_Y \overset{3\alpha k}{\not\approx} g_{X,W}(Y) \;\middle|\; Y \cap W = \emptyset, f(X \cup W)_W = f(Y \cup W)_W \right] \leq 2\epsilon^2.$$

Fix such W, X, let $G = \left\{ Y \in \binom{[N]}{\frac{9k}{10}} \;\middle|\; Y \cap W = \emptyset, f(X \cup W)_W = f(Y \cup W)_W \right\}$, and let $g = g_{X,W} : [N] \to [M]$. We want to use the last query to show that this g is in fact a global product function, i.e $f(S) \approx g(S)$ for about an ϵ fraction of $S \in \binom{[N]}{k}$.

For every set S, we say that S is bad if $f(S) \overset{5\alpha k}{\not\approx} g(S)$. Let p be the probability of a uniform S to be bad, i.e. $p = \Pr_{S \in \binom{[N]}{k}} \left[f(S) \overset{5\alpha k}{\not\approx} g(S) \right]$.

Suppose that instead of running Test 4 as is, we choose Y, V by the following process:

1. Choose a uniform $S \in \binom{[N]}{k}$.
2. Choose Y to be a uniform $\frac{9k}{10}$ subset of S.
3. Set $V = S \setminus Y$ and return (Y, V).

We suppose that if the process outputs Y such that $Y \cap W \neq \emptyset$, the test rejects. The probability of this event is less than $\frac{k^2}{N}$, and if it doesn't happen the process generates the test distribution. Therefore, the test on f using this distribution should success with probability at least $\epsilon - \frac{k^2}{N}$.

In order for Test 4 to pass, two checks must hold:

1. $f(X \cup W)_W = f(Y \cup W)_W$, equivalent to $Y \in G$.
2. $f(Y \cup V)_Y = f(Y \cup W)_Y$.

Suppose that S is bad, and we let $Y \cup V = S$ to be the sets used in the test. If $Y \notin G$, the test will fail. If $y \in G$, from the local structure, Lemma 37,

$$\Pr_Y \left[f(Y \cup W)_Y \overset{3\alpha k}{\not\approx} g(Y) \;\middle|\; Y \in G \right] \leq 2\epsilon^2.$$

If we condition on S to be bad, we restrict Y and therefore the probability of this event can increase by a factor of $\frac{1}{p}$.

$$\Pr_Y \left[f(Y \cup W)_Y \overset{3\alpha k}{\not\approx} g(Y) \;\middle|\; Y \in G, S \text{ is bad} \right] \leq \frac{1}{p} 2\epsilon^2. \tag{23}$$

Since S is bad and Y is a uniform $\frac{9k}{10}$ sized set inside S, the probability that less than $3\alpha k$ out of the $5\alpha k$ elements in which $f(S), g(S)$ differ is in Y is exponentially small.

$$\Pr_{Y\subset S}\left[f(S)_Y \overset{3\alpha k}{\approx} g(Y) \;\middle|\; S \text{ is bad}\right] \leq e^{-\frac{1}{320}\alpha k}. \tag{24}$$

The inequality is due to Chernoff bound, using Claim 11 (if D is the set of elements in which $f(S), g(S)$ differ, $f(S)_Y \overset{3\alpha k}{\approx} g(Y) \implies |Y \cap D| \leq \frac{3}{5}|D|$, in the claim we use $A = S \setminus Y$).

From equation (23), we know that with probability $1 - \frac{2\epsilon^2}{p}$, $f(Y \cup W)_Y \overset{3\alpha k}{\approx} g(Y)$. From (24), with probability $1 - e^{-\frac{1}{320}\alpha k}$, $f(S)_Y \overset{3\alpha k}{\not\approx} g(Y)$. If both holds, then $f(S)_Y = f(Y \cup V)_Y \neq f(Y \cup W)_Y$, and the test will fail. Therefore,

$$\Pr_S\left[\text{Test passes} \mid S \text{ is bad}\right] \leq e^{-\frac{1}{320}\alpha k} + \frac{2\epsilon^2}{p} \leq \frac{3\epsilon^2}{p}. \tag{25}$$

The test must pass with probability $\epsilon - \frac{k^2}{N}$,

$$\begin{aligned}\epsilon - \frac{k^2}{N} = \Pr[\text{Test passes}] =& \Pr[S \text{ is bad}]\Pr\left[\text{Test passes} \mid S \text{ is bad}\right] \\ &+ \Pr[S \text{ isn't bad}]\Pr\left[\text{Test passes} \mid S \text{ isn't bad}\right] \\ \leq& p\frac{3\epsilon^2}{p} + (1-p)\end{aligned}$$

Therefore $p = \Pr[S \text{ is bad}] \leq 1 - \epsilon + \frac{k^2}{N} + 3\epsilon^2$, which implies that at least $\epsilon - \frac{k^2}{N} - 3\epsilon^2$ of the test S are not bad, and for such sets $f(S) \overset{5\alpha k}{\approx} g(S)$. We choose $c = \frac{\delta}{5}$ so $\alpha = \frac{1}{5}\lambda$, and notice that $\epsilon - 4\epsilon^2 \leq \epsilon - \frac{k^2}{N} - 3\epsilon^2$ which finishes the proof. ◀

In the introduction, we stressed that in order to extend the restricted global structure into a global structure, the restricted global structure theorem has to be "strong", i.e. the probability of $f(Y \cup W)_Y \overset{3\alpha k}{\not\approx} g_{X,W}(Y)$ should be strictly smaller than ϵ, it is $2\epsilon^2$ in our case. If the local structure was not strong, the bound in (25) would have been larger than ϵ. This means that all the success probability of the test could come from bad sets S. From (25), we see that almost all of the success probability of the test comes from sets that are not bad, this we couldn't have deduced from the restricted structure theorem of [7].

5 Global Structure for Tuples

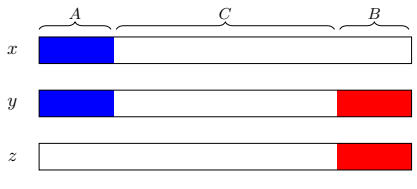
In this section we prove our main theorem – global structure for tuples. The proof uses the restricted global structure, Theorem 21. For convenience we copy the test and theorem from the introduction.

▶ **Theorem 1** (Main theorem – Global Structure for tuples). *For every $N, M > 1$, there exist small constants $c_1, c_2 > 0$ such that for every constant $\lambda > 0$ and large enough k, if $f : [N]^k \to [M]^k$ is a function that passes Test 1 with probability $\alpha_{Z(\frac{k}{10})}(f) = \epsilon \geq e^{-c_1\lambda^2 k}$, then there exist functions $(g_1, \ldots g_k)$, $g_i : [N] \to [M]$ such that*

$$\Pr_{x\in[N]^k}\left[f(x) \overset{\lambda k}{\approx} (g_1(x_1)\ldots g_k(x_k))\right] \geq c_2 \cdot \epsilon.$$

Where $\overset{\lambda k}{\approx}$ means that the strings are equal on all but at most λk coordinates.

1. Choose A, B, C to be a random partition of $[k]$, such that $|A| = |B| = t$.
2. Choose uniformly at random $x, y, z \in [N]^k$ such that $x_A = y_A$ and $y_B = z_B$.
3. Reject if $f(x)_A \neq f(y)_A$ or $f(z)_B \neq f(y)_B$, else accept.



Denote by $\alpha_{Z(t)}(f)$ the success probability of f on this test.

Test 1 "Z"-test with parameter t (3-query test).

5.1 Proof Outline

Our proof of Theorem 1 relies on Theorem 21, which gives us, for many restrictions τ, a product function g^τ that is defined on a set A of $\frac{9k}{10}$ coordinates and approximately equals f on $\mathcal{V}_\tau$. In this section we show how to stitch the restricted functions g^τ together into one global function g. The proof has three parts.

1. In Section 5.2, we show that there exist an $x \in [N]^k$ such that,
   - **a.** On at least $\Omega(\epsilon)$ of the sets A, the tuple $\tau = (A, x_A, f(x)_A)$ is excellent, and Test 1 passes with probability at least $\frac{\epsilon}{3}$.
   - **b.** Taking two such sets A_1, A_2, their functions g^{τ_1}, g^{τ_2} are similar with probability $\Omega(\epsilon^2)$.

   We start from picking x such that the test succeeds on it with probability $\Omega(\epsilon)$, and that for $\Omega(\epsilon)$ of the sets A, $\tau = (A, x_A, f(x)_A)$ satisfies the first item above. We use the third query of the test to show that each g^τ approximates f on $[k] \setminus A$ in $\Omega(\epsilon)$ of the inputs in $[N]^k$. This implies that for many different pairs τ_1, τ_2, both g^{τ_1} and g^{τ_2} are close to f on $[k] \setminus \{A_1 \cup A_2\}$ in $\Omega(\epsilon^2)$ of the inputs, which means that g^{τ_1}, g^{τ_2} are similar to each other.
2. In Section 5.3 we view this situation abstractly as *yet another* agreement question, with a different setting of parameters: given a set of direct product functions, each defined on $\frac{9}{10}k$ coordinates, such that each two are consistent with probability $\Omega(\epsilon^2)$, find a global direct product function $g = (g_1, \ldots g_k)$ that is consistent with $\Omega(\epsilon^2)$ of these functions. We show that such a g can be found, essentially proving an agreement testing theorem for this setting. This may seem circular but in fact the current setting is easier than our original problem because of the density: Since the sets are so large, every two sets intersect.

   In order to solve this agreement question, we build a graph with the functions as nodes, and connect by an edge each two consistent functions. We connect by a "weak edge" each two functions that are somewhat consistent, where we allow a larger difference between the two functions. The weak and strong edges have an "almost transitive" property, if (v_1, v_2) and (v_2, v_3) are connected by a strong edge, then almost surely (v_1, v_3) are connected by a weak edge. We use this property to show that there exist a set of vertices C of size $\Omega(\epsilon^2)$ that is almost a clique, i.e. almost every two functions in C are consistent. We build the global function by taking the plurality over C, and show that it is close to most functions in C.
3. Lastly, in Section 5.4, we connect the two previous items. The functions g^τ for $\tau = (A, x, f(x)_A)$ from the first item are each defined on $\frac{9k}{10}$ coordinates, and each two are similar with probability $\Omega(\epsilon^2)$. This means that they satisfies the conditions of the second item, and there exists a global function g defined on all $[k]$ that is close to $\Omega(\epsilon^2)$ of them. We recall that on Section 5.2 we showed that each g^τ is close to f on $\Omega(\epsilon)$ fraction of the inputs, and conclude that the global function g is also close to f on $\Omega(\epsilon)$ fraction of the input, which finishes the proof.

5.2 Consistency Between Restricted Global Functions

From Theorem 21, we know with probability $1-\epsilon^2$ a good $\tau \sim \mathcal{D}$ is excellent, and for each excellent τ there exist a local direct product function $g^\tau = (g_1^\tau, \ldots, g_{\frac{9k}{10}}^\tau)$ that equals f on $\mathcal{V}_\tau$.

▶ **Definition 38.** For every $x \in [N]^k$, let $\mathcal{A}_x$ be the set of subsets $A \subset [k]$ of size $\frac{k}{10}$ such that:

1. Fixing A, x, $\Pr_{y,B,z}[\text{Test 1 passed}] \geq \frac{\epsilon}{3}$.
2. $\tau = (A, x_A, f(x)_A)$ is excellent.

▶ **Definition 39.** Let $\tau_1 = (A_1, r_1, \gamma_1), \tau_2 = (A_2, r_2, \gamma_2)$ be two excellent tuples, we say that g^{τ_1}, g^{τ_1} are consistent if for a uniform $i \notin A_1 \cup A_2$ and $u \in [N]$,

$$\Pr_{i,u}\left[g_i^{\tau_1}(u) \neq g_i^{\tau_2}(u)\right] \leq 60\lambda.$$

The main claim we prove in this section is the following,

▶ **Claim 40.** *There exist $x \in [N]^k$, such that:*

1. $\Pr_A[A \in \mathcal{A}_x] \geq \frac{\epsilon}{4}$.
2. $\Pr_{A_1,A_2 \in \mathcal{A}_x}[g^{\tau_1}, g^{\tau_1} \text{ are consistent}] \geq \frac{\epsilon^2}{32}$, *where the tuples are* $\tau_1 = (A_1, x_{A_1}, f(x)_{A_1})$ *and* $\tau_2 = (A_2, x_{A_2}, f(x)_{A_2})$.

We start from looking for a candidate $x \in [N]^k$.

▶ **Claim 41.** *Let*

$$\mathcal{X}_1 = \left\{ x \in [N]^k \;\middle|\; \Pr[\textit{Test 1 passed with } x] \geq \frac{\epsilon}{4} \right\},$$

$$\mathcal{X}_2 = \left\{ x \in [N]^k \;\middle|\; \Pr_A[A \in \mathcal{A}_x] \geq \frac{\epsilon}{8} \right\}.$$

Then

$$\mathcal{X}_1 \cap \mathcal{X}_2 \neq \emptyset.$$

Proof. Let G be the full weighted bipartite graph, with vertex sets $L = \binom{[k]}{\frac{9k}{10}}$ and $R = [N]^k$. The weight of an edge A, x equals the success probability of Test 1 given that A, x are chosen.

The expected weight of an edge is equal to the test success probability of Test 1, ϵ. For each edges with weight less than $\frac{\epsilon}{2}$, we change its weight to 0. We removed at most half of the total weight, so the expected weight of a uniform edge now is at least $\frac{\epsilon}{2}$.

All the edges that remain with positive weight are of (A, x) such that $\tau = (A, x, f(x)_A)$ is good (there may also be good tuples with weight 0, if Test 2 passed with probability larger than $\frac{\epsilon}{2}$ but Test 1 didn't). We further change to 0 the weight of all the edges A, x such that $\tau = (A, x_A, f(x)_A)$ is not excellent.

From Theorem 21, a random good $\tau \sim \mathcal{D}$ is excellent with probability $1-\epsilon^2$, and the distribution $\tau \sim \mathcal{D}$ corresponds to a uniform choice of $A \in L, x \in R$. Therefore, changing to 0 the wight over these edges means changing to 0 the weight of at most ϵ^2 of the edges in G. The maximal weight of an edge is 1, we have reduced the expected weight by at most ϵ^2. The expected weight now is more than $\frac{\epsilon}{2} - \epsilon^2 \geq \frac{\epsilon}{3}$.

Let x be the vertex with the maximal sum of weights of neighbor edges, then

$$\Pr[\text{Test 1 passed given } x] \geq \mathop{\mathbb{E}}_A[\omega(A, x)] \geq \frac{\epsilon}{3}.$$

The inequality is because we have changed to zero the weight some edges.

All edges (A, x) that still have positive weight satisfy $A \in \mathcal{A}_x$,

$$\Pr_A [A \in \mathcal{A}_x] = \Pr_A [\omega(A, x) > 0] \geq \frac{\epsilon}{3},$$

since the maximal weight an edge can have is 1.

Therefore, $x \in \mathcal{X}_1 \cap \mathcal{X}_2$. ◀

In the rest of this section we fix $x \in \mathcal{X}_1 \cap \mathcal{X}_2$, denote $\mathcal{A} = \mathcal{A}_x$ and $g_A = g^\tau = (g_1^\tau, \ldots, g_{\frac{9k}{10}}^\tau)$ for $\tau = (A, x, f(x)_A)$, and prove that it fulfills the conditions of Claim 40.

▶ **Definition 42.** An input $z \in [N]^k$ is consistent with a set $A \in \mathcal{A}$ if $f(z)_{\bar{A}} \overset{20\lambda k}{\approx} g_A(z_{\bar{A}})$. Let $\mathcal{Z}_A$ be the set of inputs that are consistent with A.

$$\mathcal{Z}_A = \left\{ z \in [N]^k \,\middle|\, f(z)_{\bar{A}} \overset{20\lambda k}{\approx} g_A(z_{\bar{A}}) \right\}.$$

▶ **Claim 43.** *For every $A \in \mathcal{A}$, $\Pr_z [z \in \mathcal{Z}_A] \geq \frac{\epsilon}{4}$.*

Proof. Assume towards contradiction that the claim does not hold, and fix a set $A \in \mathcal{A}$ such that $\Pr_z [z \in \mathcal{Z}_A] < \frac{\epsilon}{4}$.

We reach a contradiction by showing that conditioning on A, x chosen by the test, $\Pr_{y,B,z} [\text{Test 1 passes}] < \frac{\epsilon}{3}$ contradicting the fact that $A \in \mathcal{A}$.

We define the following events, under the assumption that $y_A = x_A$ and $y_B = z_B$ as in the test.

1. E_1: $f(x)_A = f(y)_A$.
2. E_2: $f(z)_B = f(y)_B$.
3. E_3: $f(y)_B \overset{\lambda k}{\approx} g_A(y_B)$.
4. E_4: $f(z)_B \overset{\lambda k}{\approx} g_A(z_B)$.
5. E_5: $z \notin \mathcal{Z}_A$.

Note that since g_A is a product function and $y_B = z_B$, E_4 can also be written as $f(z)_B \overset{\lambda k}{\approx} g_A(y_B)$. We also notice that $E_2 \wedge E_3 \implies E_4$, since we can switch $f(y)_B$ by $f(z)_B$ in E_3.

By definition, Test 1 succeeds if E_1, E_2 both happened.

$$\begin{aligned}
\Pr_{y,B} [E_1 \wedge E_2] &\leq \Pr_{y,B,z} [E_1 \wedge E_2 \wedge E_3 \wedge E_5] + \Pr_{y,B,z} [E_1 \wedge E_2 \wedge \neg E_3] + \Pr_{y,B,z} [E_1 \wedge E_2 \wedge \neg E_5] \\
&\leq \Pr_{y,B,z} [E_1 \wedge E_4 \wedge E_5] + \Pr_{y,B,z} [E_1 \wedge E_2 \wedge \neg E_3] + \Pr_{y,B,z} [E_1 \wedge E_2 \wedge \neg E_5] \\
&\leq \Pr_{y,B,z} [E_4 \mid E_5] + \Pr_{y,B,z} [\neg E_3 \mid E_1] + \Pr_{y,B,z} [\neg E_5]. \qquad (26)
\end{aligned}$$

We bound each of the three probabilities.

1. If E_5 happened, $z \notin \mathcal{Z}_A$ so $f(z)_{\bar{A}} \overset{20\lambda k}{\not\approx} g_A(z_{\bar{A}})$, let D be the set of coordinates in which $f(z)_{\bar{A}}$ and $g_A(z_{\bar{A}})$ differ

   $$D = \left\{ i \in \bar{A} \,\middle|\, f(z)_i \neq g_{A,i}(z_i) \right\}.$$

   In order to satisfy E_4, the set B should be such that $|B \cap D| \leq \lambda k$, since B is a random set of size $\frac{k}{10}$, using Claim 11

   $$\Pr_{B,y,z} [E_4 \mid E_5] \leq \Pr_B [|B \cap D| \leq \lambda k] \leq e^{-\frac{\lambda k}{60}} < \epsilon^2.$$

2. Since $A \in \mathcal{A}_x$ the tuple $(A, x_A, f(x)_A)$ is excellent, and from Theorem 21

$$\Pr_{y,B,z}[\neg E_3 \mid E_1] = \Pr_{y,B}\left[f(y)_B \overset{\lambda k}{\not\approx} g_A(y_B) \;\middle|\; f(x)_A = f(y)_A\right] \leq \epsilon^2,$$

where we use the fact that $B \subseteq \bar{A}$, therefore $f(y)_B \overset{\lambda k}{\not\approx} g_A(y_B)$ implies $f(y)_{\bar{A}} \overset{\lambda k}{\not\approx} g_A(y_{\bar{A}})$.

3. From our assumption,

$$\Pr_{y,B,z}[E_5] = \Pr_z[z \in \mathcal{Z}_A] \leq \frac{\epsilon}{4}.$$

Therefore, from (26) we get

$$\Pr_{y,B,z}[\text{Test 1 passes} \mid x, A] = \Pr_{y,B,z}[E_1 \wedge E_2] \leq \epsilon^2 + \epsilon^2 + \frac{\epsilon}{4} < \frac{\epsilon}{3},$$

contradicting $A \in \mathcal{A}$. ◀

In the introduction, we explained the difference between our restricted global structure, and the result of [7]. In our result, Theorem 21, $f(y)_{\bar{A}} \approx g^\tau(y)$ for $1-\epsilon^2$ of $y \in \mathcal{V}_\tau$, and it their result it was much less.

▶ **Claim 44.**

$$\Pr_{A_1,A_2 \in \mathcal{A}}\left[|\mathcal{Z}_{A_1} \cap \mathcal{Z}_{A_2}| \geq \frac{\epsilon^2}{32}N^k\right] \geq \frac{\epsilon^2}{32}.$$

Proof. For a uniform pair $A_1, A_2 \in \mathcal{A}$:

$$\mathop{\mathbb{E}}_{A_1,A_2}[|\mathcal{Z}_{A_1} \cap \mathcal{Z}_{A_2}|] = \sum_z \mathop{\mathbb{E}}_{A_1,A_2}[\mathbb{I}(z \in \mathcal{Z}_{A_1} \cap \mathcal{Z}_{A_2})] \geq \sum_z \Pr_{A_1}[z \in \mathcal{Z}_{A_1}]^2 \tag{27}$$

where $\mathbb{I}$ is an indicator. The last inequality holds since A_1, A_2 are independent uniform sets in $\mathcal{A}$, and the square function is convex.

From Claim 43, $\Pr_z[z \in \mathcal{Z}_A] \geq \frac{\epsilon}{4}$ for every $A \in \mathcal{A}$. Therefore, from (27) we get

$$\begin{aligned}
\mathop{\mathbb{E}}_{A_1,A_2}[|\mathcal{Z}_{A_1} \cap \mathcal{Z}_{A_2}|] &\geq \sum_z \Pr_{A_1}[z \in \mathcal{Z}_{A_1}]^2 \\
&\geq \left(\sum_z N^{-\frac{k}{2}} \Pr_{A_1}[z \in \mathcal{Z}_{A_1}]\right)^2 && \text{(Cauchy Swartz)} \\
&\geq \left(\frac{\epsilon}{4}\right)^2 N^k. && \text{(Claim 43)}
\end{aligned}$$

The maximal value of $|\mathcal{Z}_{A_1} \cap \mathcal{Z}_{A_2}|$ is N^k, therefore by averaging

$$\Pr_{A_1,A_2}\left[|\mathcal{Z}_{A_1} \cap \mathcal{Z}_{A_2}| \geq \frac{\epsilon^2}{32}N^k\right] \geq \frac{\epsilon^2}{32}.$$ ◀

▶ **Claim 45.** *If $A_1, A_2 \in \mathcal{A}$ are such that $|\mathcal{Z}_{A_1} \cap \mathcal{Z}_{A_2}| \geq \frac{\epsilon^2}{32}N^k$, then g_{A_1}, g_{A_2} are consistent, i.e. for a uniform $i \in [k] \setminus \{A_1 \cup A_2\}$ and $u \in [N]$,*

$$\Pr_{i,u}[g_{A_1,i}(u) \neq g_{A_2,i}(u)] \leq 60\lambda.$$

Proof. Let $A_1, A_2 \in \mathcal{A}$ be two sets such that $|\mathcal{Z}_{A_1} \cap \mathcal{Z}_{A_2}| \geq \frac{\epsilon^2}{32} N^k$, and let $\mathcal{Z}_{12} = \mathcal{Z}_{A_1} \cap \mathcal{Z}_{A_2}$. In order to simplify the notation, denote $S_1 = [k] \setminus A_1$, $S_2 = [k] \setminus A_2$ and $S_{12} = S_1 \cap S_2 = [k] \setminus \{A_1 \cup A_2\}$. S_{12} is the set of coordinates that both g_{A_1}, g_{A_2} are defined on, $|S_{12}| \geq 0.8k$.

For each $i \in S_{12}$, let

$$p_i = \Pr_{u \in [N]} \left[g_{A_1,i}(u) \neq g_{A_2,i}(u) \right].$$

Let $w \in [N]^{S_{12}}$ uniformly at random, and let I_i be indicator random variable for $g_{A_1,i}(w_i) \neq g_{A_2,i}(w_i)$. Each I_i equals 1 with probability p_i independently. In this notation

$$\mathop{\mathbb{E}}_{w} \left[\mathrm{dist}(g_{A_1}(w), g_{A_2}(w)) \right] = \mathbb{E} \left[\sum_{i \in S_{12}} I_i \right].$$

Assume towards contradiction that $\Pr_{i,b} \left[g_{A_1,i}(u) \neq g_{A_2,i}(u) \right] > 60\lambda$, this will imply that $\mathbb{E}_w \left[\mathrm{dist}(g_{A_1}(w), g_{A_2}(w)) \right] > 60\lambda \cdot 0.8k$.

Using Chernoff bound:

$$\Pr_{w} \left[\mathrm{dist}(g_{A_1}(w), g_{A_2}(w)) \leq 40\lambda k \right] = \Pr \left[\sum_{i \in S_{12}} I_i \leq \frac{5}{6} \mathbb{E} \left[\sum_{i \in S_{12}} I_i \right] \right] \leq e^{-\frac{1}{2}\lambda k}.$$

If instead of taking a completely uniform $w \in [N]^{S_{12}}$, we pick a random $z \in \mathcal{Z}_{12}$, and restrict it to S_{12}, getting $w = z_{S_{12}}$. The probability of any event on w can increase by a factor of at most $\frac{N^k}{|\mathcal{Z}_{12}|} \leq \frac{32}{\epsilon^2}$,

$$\Pr_{z \in \mathcal{Z}_{12}} \left[\mathrm{dist}(g_{A_1}(z_{S_{12}}), g_{A_2}(z_{S_{12}})) \leq 40\lambda k \right] \leq \frac{32}{\epsilon^2} e^{-\frac{1}{2}\lambda k} < \frac{1}{2}. \tag{28}$$

By the definition of $\mathcal{Z}_{A_1}, \mathcal{Z}_{A_2}$, each input $z \in \mathcal{Z}_{12}$ satisfies both $f(z)_{S_1} \overset{20\lambda k}{\approx} g_{A_1}(z_{S_1})$ and $f(z)_{S_2} \overset{20\lambda k}{\approx} g_{A_2}(z_{S_2})$ which implies $g_{A_1}(z_{S_{12}}) \overset{40\lambda k}{\approx} g_{A_2}(z_{S_{12}})$ with probability 1, which contradicts (28). ◀

Combining the last two claims, we prove Claim 40.

5.3 Agreement Theorem in the Dense Case

In this section, we present and prove an abstract problem that will later be used to create the global product function. Given a collection of local functions $\mathcal{F} = \{f_S\}_{S \in \binom{[k]}{\frac{9k}{10}}}$, such that for each $S \in \binom{[k]}{\frac{9k}{10}}$, $f_S : S \to \Sigma$, can we deduce from the agreement of f_S the existence of a single global function $g : [k] \to \Sigma$ that is close to many f_S?

We need to define what exactly agreement means in the case of $\mathcal{F}$, as it is not the setting on which we previously defined agreement on. In order to do so, we assume that we have a bounded distance measure on Σ, i.e. for every $\sigma_1, \sigma_2 \in \Sigma$, $\mathrm{dist}(\sigma_1, \sigma_2) \in [0, 1]$.

▶ **Definition 46.** The difference between $f_{S_1}, f_{S_2} \in \mathcal{F}$, denoted by $\Delta(f_{S_1}, f_{S_2})$ is defined by

$$\Delta(f_{S_1}, f_{S_2}) = \mathop{\mathbb{E}}_{i \in S_1 \cap S_2} [\mathrm{dist}(f_{S_1}(i), f_{S_1}(i))].$$

The difference between $f_S \in \mathcal{F}$ to a function $g : [k] \to \Sigma$ is defined by,

$$\Delta(f_S, g) = \mathop{\mathbb{E}}_{i \in S} [\mathrm{dist}(f_S(i), g(i))].$$

Note that the difference defined above is not a distance, it may be that $\Delta(f_{S_1}, f_{S_2}) = 0$ for $S_1 \neq S_2$.

Now we are ready to define the agreement, notice that since we are talking on an agreement inside a function set $\mathcal{F}$, the definition is different. The general idea is the same – we check for the agreement of two random elements in $\mathcal{F}$ according to some distribution.

▶ **Definition 47.** The agreement of the collection of local functions $\mathcal{F}$ regarding the uniform distribution with parameter α, denoted by $\text{agree}_\alpha(\mathcal{F})$ is defined by,

$$\text{agree}_\alpha(\mathcal{F}) = \Pr_{f_{S_1}, f_{S_2} \in \mathcal{F}}[\Delta(f_{S_1}, f_{S_2}) < \alpha].$$

▶ **Theorem 48.** *For every small constant $\alpha \in (0,1)$ and $\nu > e^{-\frac{1}{3}\alpha^2 k}$, if a collection of local functions $\mathcal{F}$ has $\text{agree}_\alpha(\mathcal{F}) > \nu$, then there exists a global function $g : [k] \to \Sigma$ such that*

$$\Pr_{S \in \binom{[k]}{\frac{9k}{10}}}[\Delta(f_S, g) \leq 300\alpha] \geq \frac{1}{4}\nu.$$

In order to prove the theorem, it is helpful to look at the elements $S \in \binom{[k]}{\frac{9k}{10}}$ as vertices in a graph. Let $\mathcal{G} = (\mathcal{V}, E_S \cup E_W)$ to be the graph with the vertex set $\mathcal{V} = \binom{[k]}{\frac{9k}{10}}$, and two edge sets, weak edges and strong edges.

▶ **Definition 49.** For every two sets $S_1, S_2 \in \mathcal{V}$:

1. S_1, S_2 are connected by a strong edge, denoted by $S_1 - S_2$, if $\Delta(f_{S_1}, f_{S_2}) < \alpha$.
2. S_1, S_2 are connected by a weak edge, denoted by $S_1 \sim S_2$, if $\Delta(f_{S_1}, f_{S_2}) < 60\alpha$.

We want to find a subset of vertices that is close to a clique in $\mathcal{G}$, such subset will allow us to define a global function g. We start by showing that there exist many vertices of high degree in $\mathcal{G}$.

▶ **Claim 50.** *Exists a set $\mathcal{S} \subset \mathcal{V}$ of measure at least $\frac{\nu}{2}$, such that for every $S \in \mathcal{S}$*

$$\Pr_{S' \in \mathcal{V}}[S - S'] \geq \frac{1}{2}\nu.$$

Proof. Let

$$\mathcal{S} = \left\{ S \subseteq \mathcal{V} \;\middle|\; \Pr_{S'}[S - S'] \geq \frac{1}{2}\nu \right\}.$$

By averaging

$$\begin{aligned}
\nu &\leq \Pr_{S_1, S_2}[S_1 - S_2] \\
&\leq \Pr_{S_1}[S_1 \in \mathcal{S}] \Pr_{S_1, S_2}[S_1 - S_2 \mid S_1 \in \mathcal{S}] + \Pr_{S_1}[S_1 \notin \mathcal{S}] \Pr_{S_1, S_2}[S_1 - S_2 \mid S_1 \notin \mathcal{S}] \\
&\leq \Pr_{S_1}[S_1 \in \mathcal{S}] + \frac{1}{2}\nu\left(1 - \Pr_{S_1}[S_1 \in \mathcal{S}]\right).
\end{aligned}$$

Then $\Pr_{S_1}[S_1 \in \mathcal{S}] \geq \frac{1}{2}\nu$. ◀

Strong connectivity is not transitive, but we can have an "almost transitive" property by considering both strong and weak edges.

▶ **Claim 51.** *For $S, S_1, S_2 \in \mathcal{V}$ uniformly and independently,*

$$\Pr_{S,S_1,S_2}[S - S_1, S - S_2, S_1 \not\sim S_2] \leq 2e^{-\alpha^2 k}.$$

Proof. Fix $S_1, S_2 \in \mathcal{V}$ to be two vertices such that $S_1 \not\sim S_2$ (if there are no such vertices, the probability is 0 and we are done).

For every $S \in \mathcal{V}$, we define by d_i, d_i^1, d_i^2 the following distances:

1. For each $i \in S_1 \cap S_2$, $d_i = \text{dist}(f_{S_1}(i), f_{S_2}(i))$.
2. For each $i \in S \cap S_1$, $d_i^1 = \text{dist}(f_S(i), f_{S_1}(i))$.
3. For each $i \in S \cap S_2$, $d_i^2 = \text{dist}(f_S(i), f_{S_2}(i))$.

By the triangle inequality, for every $i \in S \cap S_1 \cap S_2$, $d_i \leq d_i^1 + d_i^2$, therefore for every such i, $\max\{d_i^1, d_i^2\} \geq \frac{d_i}{2}$.

Since $S_1 \not\sim S_2$, we know that $\mathbb{E}_{i \in S_1 \cap S_2}[d_i] \geq 60\alpha$, if we look on the sum $\sum_{i \in S_1 \cap S_2} d_i \geq \frac{8k}{10} 60\alpha$ (because $|S_1 \cap S_2| \geq \frac{8}{10}k$). If $S - S_1, S - S_2$, then $\max\{\mathbb{E}_{i \in S \cap S_1}[d_i^1], \mathbb{E}_{i \in S \cap S_2}[d_i^2]\} \leq \alpha$, which means that $\max\{\sum_{i \in S \cap S_1} d_i^1, \sum_{i \in S \cap S_2} d_i^2\} \leq \frac{9}{10}\alpha k$ (we switched expectation in a sum, $|S \cap S_1| \leq \frac{9k}{10}$).

$$\max\left\{\sum_{i \in S \cap S_1} d_i^1, \sum_{i \in S \cap S_2} d_i^2\right\} \geq \frac{1}{2} \sum_{i \in S \cap S_1 \cap S_2} \max\left\{d_i^1, d_i^2\right\} \geq \frac{1}{4} \sum_{i \in S \cap S_1 \cap S_2} d_i \tag{29}$$

The first inequality is since taking the maximum over every i can increase the total sum in a factor of 2 at most from taking maximum of the sum. The second inequality is since $\max\{d_i^1, d_i^2\} \geq \frac{d_i}{2}$.

Notice that the last expression is independent of the function f_S, and depends only on the set S. Let X_S be the random variable $X_S = \frac{1}{4}\sum_{i \in S \cap S_1 \cap S_2} d_i$ for a uniform $S \in \mathcal{V}$. Since the set S is a uniform $\frac{9k}{10}$ sized subset of $[k]$, $\mathbb{E}_S[X_S] = \frac{9}{10}\sum_{i \in S_1 \cap S_2} d_i \geq \frac{9}{10}\frac{8}{10}60\alpha k$. For $S \in \mathcal{V}$ such that $X_S > \alpha k$, by (29) it means that S is not strongly connected to one of S_1, S_2.

To finish the proof, we need to show that $X_S \leq \alpha k$ for very few $S \in \mathcal{V}$. Let D contain the $\frac{k}{3}$ indices $i \in S_1 \cap S_2$ with the largest d_i. Obviously $\sum_{i \in D} d_i \geq \frac{1}{3}\sum_{i \in S_1 \cap S_2} d_i \geq 16\alpha k$. In order of $X_S \leq \alpha k$, the sum over $i \in D \cap S$ should satisfy, $\sum_{i \in D \cap S} d_i \leq 4\alpha k$. By Claim 52, this happens with probability less than $2e^{-\alpha^2 k}$.

Therefore, for every $S_1 \not\sim S_2$, the probability of a uniform $S \in \mathcal{V}$ to be strongly connected to both is at most $2e^{-\alpha^2 k}$. This is true for every $S_1 \not\sim S_2$, it is also true for a random pair. ◀

▶ **Claim 52** (Fixed sized Chernoff bound)**.** *For every constant $\alpha \in (0,1)$, let $k \in \mathbb{N}$ be a large enough integer, $D \subset [k]$ a subset of size at most $\frac{k}{3}$, and for every $i \in D$ let $d_i \in [0,1]$ be constants such that $\sum_{i \in D} d_i > 4\alpha k$.*

Let $S \subset [k]$ be a random subset of size exactly $\frac{9k}{10}$, then

$$\Pr_S\left[\sum_{i \in S \cap D} d_i \leq \alpha k\right] \leq 2e^{-\alpha^2 k}. \tag{30}$$

Proof. For a set S that is chosen by putting each $i \in [k]$ in S with probability $\frac{9}{10}$ independently, Chernoff bound gives us the required bound easily. Because S has a fixed size, we need to work a little harder.

For each $i \in D$, let S be a uniform set in $\binom{[k]}{\frac{9k}{10}}$, and let I_i be,

$$I_i = \begin{cases} d_i & i \in S \\ 0 & i \notin S \end{cases}.$$

In this notation, $\sum_{i\in S\cap D} d_i = \sum_{i\in D} I_i$. The random variables I_i are not independent, we define the independent random variables J_i,

$$J_i = \begin{cases} d_i & \text{w.p } \frac{1}{2} \\ 0 & \text{w.p } \frac{1}{2} \end{cases}.$$

Since $|D| = \frac{k}{3}$, and S is a uniform $\frac{9}{10}k$ sized subset of $[k]$, even conditioning on all other $j \in D \setminus \{i\}$ to be in S, the probability of i to be in S is at least $\frac{1}{2}$.

$$\Pr\left[I_i = d_i \mid \forall j \in D \setminus \{i\}, I_j > 0\right] \geq \Pr\left[J_i = d_i\right]. \tag{31}$$

So a lower bound for J_i implies a lower bound for I_i.

The random variables J_i satisfies $\mathbb{E}\left[\sum_{i\in D} J_i\right] = \frac{1}{2}\sum_{i\in D} d_i \geq 2\alpha k$.

$$\begin{aligned}
\Pr_S\left[\sum_{i\in S\cap D} d_i \leq \alpha k\right] &= \Pr_{I_i}\left[\sum_{i\in D} I_i \leq \alpha k\right] \\
&\leq \Pr_{J_i}\left[\sum_{i\in D} J_i \leq \alpha k\right] \\
&\leq \Pr_{J_i}\left[\left|\sum_{i\in D} J_i - \mathbb{E}\left[\sum_{i\in D} J_i\right]\right| \geq \alpha k\right] && \text{(Chernoff bound)} \\
&\leq 2e^{-\alpha^2 k}.
\end{aligned}$$

◀

From the last two claims, Claim 50 and Claim 51, conclude that there is a high degree vertex in $\mathcal{V}$ that its neighbors almost form a clique.

▶ **Claim 53.** *There exists a set $S \in \mathcal{S}$ such that*

$$\Pr_{S_1,S_2\in\mathcal{V}}\left[S_1 \sim S_2 \mid S_1 - S, S_2 - S\right] \geq 1 - \alpha.$$

Proof. From Claim 50, we know that if we choose $S, S_1, S_2 \in \mathcal{V}$ independently,

$$\Pr_{S,S_1,S_2}\left[S \in \mathcal{S}, S - S_1, S - S_2\right] \geq \Pr_S\left[S \in \mathcal{S}\right] \Pr_{S,S_1}\left[S - S_1 \mid S \in \mathcal{S}\right]^2 \geq \left(\frac{\nu}{2}\right)^3.$$

From Claim 51, on the same distribution

$$\Pr_{S,S_1,S_2}\left[S - S_1, S - S_2, S_1 \not\sim S_2\right] \leq 2e^{-\alpha^2 k}.$$

Therefore

$$\Pr_{S,S_1,S_2}\left[S_1 \not\sim S_2 \mid S \in \mathcal{S}, S - S_1, S - S_2\right] \leq \left(\frac{2}{\nu}\right)^3 2e^{-\alpha^2 k} < \alpha.$$

The last inequality is since $\log\left(\frac{1}{\nu}\right) \leq \frac{1}{3}\alpha^2 k$.

From averaging, there must be $S \in \mathcal{S}$ that achieves this bound. ◀

Proof of Theorem 48. Let $\tilde{S} \in \mathcal{S}$ be the vertex promised from Claim 53, and denote by C its strong neighbors,

$$C = \left\{S \in \binom{[k]}{\frac{9k}{10}} \;\middle|\; S - \tilde{S}\right\},$$

since $\tilde{S} \in \mathcal{S}$, the measure of C is at least $\frac{\nu}{2}$. From the claim we also know that $\Pr_{S_1,S_2\in C}[S_1 \not\sim S_2] \leq \alpha$, so almost every two sets in C have small difference.

The global function $g(i)$ is defined to be $\beta \in \Sigma$ that is closest to $f_S(i)$ over all $S \in C$ that contains i,

$$\forall i \in [k], \quad g(i) = \operatorname*{argmin}_{\beta\in\Sigma} \left\{ \mathop{\mathbb{E}}_{S\in C \text{ s.t. } i\in S} [\operatorname{dist}(f_S(i), \beta)] \right\}.$$

If there is no $S \in C$ such that $i \in S$, we define $g(i)$ to an arbitrary value.

We notice that for every i, by definition

$$\Pr_{S\in C \text{ s.t. } i\in S} [\operatorname{dist}(f_S(i), g(i))] \leq \Pr_{S_1,S_2\in C \text{ s.t. } i\in S_1,S_2} [\operatorname{dist}(f_{S_1}(i), f_{S_2}(i))]. \tag{32}$$

We know that $S_1, S_2 \in C$ are weakly connected with probability at least $1 - \alpha$, which means that the difference between their functions is small.

$$\begin{aligned} \mathop{\mathbb{E}}_{S_1,S_2\in C} [\Delta(f_{S_1}, f_{S_2})] \leq & 1 \cdot \Pr_{S_1,S_2\in C} [S_1 \not\sim S_2] + \mathop{\mathbb{E}}_{S_1,S_2\in C} [\Delta(f_{S_1}, f_{S_2}) \mid S_1 \sim S_2] \\ \leq & \alpha + 60\alpha \leq 61\alpha. \end{aligned}$$

By the definition of difference, we get that,

$$\begin{aligned} 61\alpha \geq & \mathop{\mathbb{E}}_{S_1,S_2\in C} [\Delta(f_{S_1}, f_{S_2})] \\ \geq & \mathop{\mathbb{E}}_{S_1,S_2\in C, i\in S_1\cap S_2} [\operatorname{dist}(f_{S_1}(i), f_{S_2}(i))]. \end{aligned} \tag{33}$$

Notice that the distribution over i in this expression is not uniform, we define formally the distributions over i that we use.

1. Let $\mathcal{D}_1 : [k] \to [0, 1]$ be the distribution that picks $S \in C$ uniformly, then $i \in S$.
2. Let $\mathcal{D}_2 : [k] \to [0, 1]$ be the distribution that picks $S_1, S_2 \in C$ uniformly, then $i \in S_1 \cap S_2$ (as $|S_i| = \frac{9k}{10}$ there is always such i).

Using this definition, (33) can also be written as

$$\mathop{\mathbb{E}}_{i\sim\mathcal{D}_2, S_1,S_2\in C} [\operatorname{dist}(f_{S_1}(i), f_{S_2}(i)) \mid i \in S_1 \cap S_2] \leq 61\alpha. \tag{34}$$

To prove the theorem, we need to prove (34) when $i \sim \mathcal{D}_1$. First, we show that the distributions $\mathcal{D}_1, \mathcal{D}_2$ are close to each other. In order to do so, we define the following set, D,

$$D = \left\{ i \in [k] \;\middle|\; \Pr_{S\in C} [i \in S] < \frac{1}{2} \right\}.$$

By Claim 54, the set D is small $|D| \leq 4\alpha k$. For each $i \notin D$, $\Pr_{S\in C} [i \in S] \in \left[\frac{1}{2}, 1\right]$ which means that for every $i \notin D$,

$$\Pr_{j\sim\mathcal{D}_1} [j = i \mid j \notin D] \leq 2 \Pr_{j\sim\mathcal{D}_2} [j = i \mid j \notin D]. \tag{35}$$

Using (35), (34) and (32), we show that the expected difference between g and f_S for a

random $S \in C$ is small,

$$\begin{aligned}
\mathop{\mathbb{E}}_{S\in C}[\Delta(f_S,g)] &= \mathop{\mathbb{E}}_{S\in C, i\in S}[\mathrm{dist}(f(i),g(i))] && \text{(by definition of } \mathcal{D}_1\text{)}\\
&= \mathop{\mathbb{E}}_{i\sim\mathcal{D}_1, S\in C}[\mathrm{dist}(f_{S_1}(i),g(i)) \mid i\in S] && \text{(by (32))}\\
&\le \mathop{\mathbb{E}}_{i\sim\mathcal{D}_1, S_1,S_2\in C}[\mathrm{dist}(f_{S_1}(i),f_{S_2}(i)) \mid i\in S_1\cap S_2]\\
&\le \Pr_{i\sim\mathcal{D}_1}[i\in D] + \mathop{\mathbb{E}}_{i\sim\mathcal{D}_1, S_1,S_2\in C}[\mathrm{dist}(f_{S_1}(i),f_{S_2}(i)) \mid i\in S_1\cap S_2\setminus D]\\
& && \text{(by (35))}\\
&\le 4\alpha + 2\mathop{\mathbb{E}}_{i\sim\mathcal{D}_2, S_1,S_2\in C}[\mathrm{dist}(f_{S_1}(i),f_{S_2}(i)) \mid i\in S_1\cap S_2\setminus D] && \text{(by (34))}\\
&\le 4\alpha + 2\cdot\frac{61\alpha}{1-4\alpha} \le 150\alpha. && (36)
\end{aligned}$$

Equation (32) holds for every $i \in [k]$, therefore it holds for expectation over i under any distribution. The last inequality holds because of (34), and because if we condition on $i \notin D$ we can increase the probability by a factor of at most $\Pr_{i\sim\mathcal{D}_2}[i \notin D]$, which is small.

The only thing left now is a Markov argument, if $\mathbb{E}_{S\in C}[\Delta(f_S,g)] \le 150\alpha$, then at least half of the sets $S \in C$ satisfies $\Delta(f_S,g) \le 300\alpha$, since the measure of C is $\frac{\nu}{2}$, the measure of half of C is $\frac{\nu}{4}$ and we are done. ◀

▶ **Claim 54.** *Let $C \subset \binom{[k]}{\frac{9k}{10}}$ a subset of fraction size $\frac{\nu}{2}$, then the number of indices $i \in k$ such that $\Pr_{S\in C}[i \in S] \le \frac{1}{2}$ is at most $4\alpha k$.*

Proof. Let $D \subset [k]$ be this set of indices

$$D = \left\{ i\in[k] \;\middle|\; \Pr_{S\in C}[i\in S] \le \frac{1}{2} \right\}.$$

If we pick a completely uniform $S' \in \binom{[k]}{\frac{9}{10}k}$,

$$\mathop{\mathbb{E}}_{S'}[|S'\cap D|] = \frac{9}{10}|D|.$$

From Chernoff, using Claim 11 with $A = [k]\setminus S'$, (if $|D| \ge \frac{k}{3}$, the probability is even smaller)

$$\Pr_{S'}\left[|S'\cap D| \le \frac{2}{3}|D|\right] \le e^{-\frac{|D|}{45}}.$$

If we pick a uniform subset in $S \in C$, instead of a completely uniform set:

$$\Pr_{S\in C}\left[|S\cap D| \le \frac{2}{3}|D|\right] \le \frac{2}{\nu}e^{-\frac{|D|}{45}}.$$

From the definition of D, for each $i \in D$, $\Pr_{S\in C}[i \in S] \le \frac{1}{2}$, so of course

$$\mathop{\mathbb{E}}_{S\in C}[|S\cap D|] \le \frac{1}{2}|D|.$$

From averaging

$$\Pr_{S\in C}\left[|S\cap D| \le \frac{2}{3}|D|\right] \ge \frac{1}{4}.$$

This implies that $\frac{2}{\nu}e^{-\frac{|D|}{45}} \ge \frac{1}{4}$, which means that $|D| \le 4\alpha k$ (recall that $\nu > e^{-\frac{1}{150}\alpha k}$). ◀

5.4 Direct Product Function Inputs

In Section 5.2 we proved Claim 40, let $\mathcal{A} = \mathcal{A}_x$ for this input x. From the claim, we know that for each $A \in \mathcal{A}$ there exists a direct product function g_A such that,

$$\Pr_{A_1,A_2\in\mathcal{A}}[g_{A_1}, g_{A_1} \text{ are consistent}] \geq \frac{\epsilon^2}{32}.$$

We want to use Theorem 48 in order to build a global direct product function. For every $A \in \mathcal{A}$, the direct product function $g_A = (g_{A,1}, \dots g_{A,\frac{9k}{10}})$, $g_{A,i} : [N] \to [M]$ can also be written as $f_S : S \to \Sigma$, where $S = [k] \setminus A$, and $\Sigma = [M]^N$. For every $i \in S$, $f_S(i)$ is the truth table of $g_{A,i}$. The distance measure in Σ is the normalized hamming distance between two strings in $[M]^N$, i.e.

$$\text{dist}(\sigma_1, \sigma_2) = \Pr_{u\in[N]}[\sigma_1(u) \neq \sigma_2(u)].$$

From the definition of consistent, for every consistent A_1, A_2, the functions f_{S_1}, f_{S_2} satisfy $\Delta(f_{S_1}, f_{S_2}) < 60\lambda$.

For every $A \notin \mathcal{A}$, we define a "fake" function f_S for $S = [k] \setminus A$, and assume that its outputs are at distance 1 from any other outputs, i.e. for every $S' \in \binom{[N]}{k}, i \in S \cap S'$, $\text{dist}(f_S(i), f_{S'}(i)) = 1$.

Let $\mathcal{F}$ be the collection of local functions $\{f_S\}_{S\in\binom{[N]}{k}}$ that we have just defined, let $\alpha = 60\lambda$ and $\nu = \left(\frac{\epsilon}{4}\right)^2 \frac{\epsilon^2}{32} = \frac{\epsilon^4}{512}$.

$$\text{agree}_\alpha(\mathcal{F}) = \Pr_{A_1,A_2}[A_1, A_2 \in \mathcal{A}, A_1, A_2 \text{ are consistent}] \geq \left(\frac{\epsilon}{4}\right)^2 \frac{\epsilon^2}{32} = \nu.$$

In order of the theorem to hold, we need $\nu = \frac{\epsilon^4}{32} = \frac{1}{32}e^{-4c_1\lambda^2 k}$ to satisfy $\nu > e^{-\frac{1}{3}\alpha^2 k} = e^{-\frac{1}{3}(60\lambda)^2 k}$, this holds for a small enough c_1.

By Theorem 48, there exists a product function $g' : [k] \to \Sigma$ which is close to $\frac{\nu}{4}$ of the functions f_S. Translating it back to our setting, we can write g' as $g = (g_1, \dots g_k), g_i : [N] \to [M]$, and a set $\mathcal{A}^*$ of size $\frac{\nu}{4} = \frac{1}{2048}\epsilon^4$, such that for each $A \in \mathcal{A}^*$,

$$\Pr_{i\in\bar{A},u\in[N]}[g_i(u_i) \neq g_{A,i}(u_i)] \leq 300\alpha = 18000\lambda.$$

For simplicity of notations, let $\delta = 300\alpha$. Notice that by our definition, for each $A \notin \mathcal{A}$ the function g_A never agrees with any other function, therefore $\mathcal{A}^* \subset \mathcal{A}$.

▶ **Definition 55.** An input z is consistent with a set $A \in \mathcal{A}^*$ with respect to the product function g, denoted by $z \in \mathcal{Z}_A^g$, if $z \in \mathcal{Z}_A$, and $g_A(z_{\bar{A}}) \overset{2\delta k}{\approx} g(z)_{\bar{A}}$.

▶ **Claim 56.** *For each $A \in \mathcal{A}^*$,*

$$\Pr_{z\in[N]^k}[z \in \mathcal{Z}_A^g] \geq \frac{\epsilon}{8}.$$

Proof. Fix $A \in \mathcal{A}^*$, for each $i \in \bar{A}$, denote by p_i the probability of g, g_A to differ on the ith coordinate, $p_i = \Pr_{u\in[N]}[g_{A,i}(u) \neq g_i(u)]$, from Theorem 48 $\mathbb{E}_{i\in\bar{A}}[p_i] \leq \delta$.

Let I_i be the indicator random variable that equals 1 with probability p_i independently for each i. For a uniform $z \in [N]^k$,

$$\mathbb{E}_{z\in[N]^k}[\text{dist}(g_A(z), g(z)_{\bar{A}})] = \sum_{i\in\bar{A}} I_i.$$

Using Chernoff

$$\Pr_z\left[g_A(z_{\bar{A}}) \overset{2\delta k}{\not\approx} g(z)_{\bar{A}}\right] \leq \Pr\left[\sum_{i\in[k]\setminus A} I_i \geq 2\,\mathbb{E}\left[\sum_{i\in[k]\setminus A} I_i\right]\right] \leq e^{-\frac{1}{9}\delta k} \leq \frac{\epsilon}{8}.$$

We know that $A \in \mathcal{A}$, therefore $\Pr_z[z \in \mathcal{Z}_A] \geq \frac{\epsilon}{4}$, therefore

$$\Pr_z[z \in \mathcal{Z}_A^g] \geq \Pr_z\left[z \in \mathcal{Z}_A, g_A(z_{\bar{A}}) \overset{2\delta k}{\approx} g(z)_{\bar{A}}\right] \geq \frac{\epsilon}{4} - \frac{\epsilon}{8} \geq \frac{\epsilon}{8}.$$

◀

▶ **Claim 57.** *If $z \in [N]^k$ is satisfies $z \in \mathcal{Z}_A^g$ for more than $\frac{\epsilon}{16}$ fraction of the sets $A \in \mathcal{A}^*$, then $f(z) \overset{3\delta k}{\approx} g(z)$.*

Proof. Fix $z \in [N]^k$ such that $z \in \mathcal{Z}_A^g$ for more than $\frac{\epsilon}{16}$ fraction of the sets $A \in \mathcal{A}^*$.

Assume towards contradiction that $f(z) \overset{3\delta k}{\not\approx} g(z)$, and denote by $D \subset [k]$ the set of coordinates in which they differ

$$D = \{i \in [k] \mid \ f(z)_i \neq g(z)_i\}.$$

For each A such that $z \in \mathcal{Z}_A^g$, by definition $g(z)_{\bar{A}} \overset{2\delta k}{\approx} g_A(z_{\bar{A}})$. Since $z \in \mathcal{Z}_A$, we also know that $g_A(z_{\bar{A}}) \overset{20\lambda k}{\approx} f(z)_{\bar{A}}$. Using both,

$$g(z)_{\bar{A}} \overset{2\delta k+20\lambda k}{\approx} f(z)_{\bar{A}}.$$

By the definition of D, this implies that $|\bar{A} \cap D| \leq 2\delta k + 20\lambda k \leq 2.1\delta k$, the rest of D must be in A, $|A \cap D| \geq |D| - 2.1\delta k$. According to our assumption, $|D| \geq 3\delta k$, which implies that $|A \cap D| \geq \frac{1}{4}|D|$.

From the previous paragraph, all sets A such that $z \in \mathcal{Z}_A^g$ satisfies $|A \cap D| \geq \frac{1}{4}|D|$, and there are $\frac{\epsilon}{16}|\mathcal{A}^*|$ such sets.

From Claim 11, we know that for a random set $A \subset [k]$ of size $\frac{1}{10}k$,

$$\Pr_A\left[|D \cap A| \geq \frac{1}{4}|D|\right] \leq e^{-150\lambda k}.$$

The set $\mathcal{A}^*$ has measure $\frac{\epsilon^4}{2048}$,in order to satisfy the requirements $\frac{\epsilon}{16}\frac{\epsilon^4}{2048} < e^{-150\lambda k}$, and we reach a contradiction. ◀

The previous claims practically finishes the proof

Proof of Theorem 1. From Claim 56, each $A \in \mathcal{A}^*$ satisfies $|\mathcal{Z}_A^g| \geq \frac{\epsilon}{8}N^k$, therefore

$$\mathbb{E}_z[|\{A \in \mathcal{A}^* \mid \ z \in \mathcal{Z}_A^g\}|] = \sum_{A\in\mathcal{A}^*} \mathbb{E}_z[\mathbb{I}(z \in \mathcal{Z}_A^g)] = \frac{1}{8}\epsilon|\mathcal{A}^*|.$$

From averaging, a uniform $z \in [N]^k$ satisfies $|\{A \in \mathcal{A}^* \mid \ z \in \mathcal{Z}_A^g\}| \geq \frac{1}{16}\epsilon|\mathcal{A}^*|$ with probability at least $\frac{1}{16}\epsilon$. Using Claim 57, each such input z satisfies $f(z) \overset{3\delta k}{\approx} g(z)$. We chose $\delta = 300\alpha = 18000\lambda$, in order to get that $f(z) \overset{\lambda' k}{\approx} g(z)$ we just need to choose small enough c_1, and substitute $\lambda' = \frac{1}{18000}\lambda$ in the proof. ◀

6 Lower Bounds for Approximate Equality

Our direct product theorem states that if a function $f : [N]^k \to [M]^k$ passes Test 1 with $t = \frac{k}{10}$ with probability $\epsilon > e^{-c_1\lambda^2 k}$, i.e. $\alpha_{Z(\frac{k}{10})}(f) > e^{-c_1\lambda^2 k}$, then there exists a direct product function $g = (g_1, \dots g_k)$ such that

$$\Pr_{x\in[N]^k}\left[f(x) \overset{\lambda k}{\approx} g(x)\right] \geq \Omega(\epsilon).$$

Ideally, we want the stronger conclusion that

$$\Pr_{x\in[N]^k}\left[f(x) = g(x)\right] \geq \Omega(\epsilon).$$

i.e., replacing approximate equality with equality.

In the introduction there is an example explaining why approximate equality is necessary for f such that $\alpha_{Z(\frac{k}{10})}(f) \geq e^{-\delta k}$. In this section, we show two extensions.

1. We generalize Test 1 with intersection size t to Test 5 with two intersection parameters $t_1, t_2 \in \mathbb{N}, t_1 + t_2 \leq k$, and show a lower bound for Test 5 with every such t_1, t_2 (Test 5 with $t_1 = t_2$ is equivalent to Test 1).
2. We analyze the triangle test, Test 6, and give a lower bound for this test.

▶ **Definition 58.** We say that functions $f_1, f_2 : [N]^k \to [M]^k$ are (ϵ, δ) close, if

$$\Pr_{x\in[N]^k}\left[f_1(x) \overset{\delta k}{\approx} f_2(x)\right] \geq \epsilon.$$

A function $f : [N]^k \to [M]^k$ is (ϵ, δ) far from direct product, if there is no direct product function $g = (g_1, \dots, g_k) : [N]^k \to [M]^k$ that is (ϵ, δ) close to f.

Recall $w \overset{t}{\approx} w'$ if w, w' are equal in all but t of the coordinates.

In this notation, Theorem 1 states that if $\alpha_{Z(\frac{k}{10})}(f) = \epsilon > e^{-c_1\lambda^2 k}$, then f is $(\Omega(\epsilon), \lambda)$ close to a direct product function. We are interested to know if it is possible to have a direct product theorem such that f is $(\Omega(\epsilon), 0)$ close to a direct product function.

Let h be the function from Example 3 in the introduction, it satisfies $\alpha_{Z(\frac{k}{10})}(h) = \epsilon > e^{-c_1\lambda^2 k}$, but is $(c \cdot \epsilon, \lambda)$-far from a direct product function for any constant c. Therefore, it is not true that $\alpha_{Z(\frac{k}{10})}(h) > e^{-c_1\lambda^2 k}$ implies $(\Omega(\epsilon), 0)$ close to a direct product function.

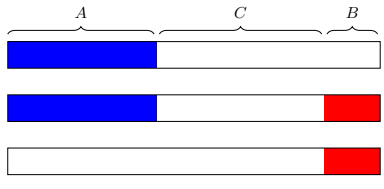
h is a direct product function with noise, on each input $x \in [N]^k$, $h(x)$ is corrupted on λk coordinates. The direct product test with $t = \frac{k}{10}$ does not check all the coordinates of each input, so with probability $e^{-\lambda\delta k}$, non of the corrupted coordinates are checked. However, if we change the parameters of the test from $t = \frac{k}{10}$ to $t = \frac{k}{2}$, in which all coordinates of the input y are checked, the function h no longer passes the test.

Is it possible to prove $(\Omega(\epsilon), 0)$ close for Test 1 with $t = \frac{1}{2}$? the answer is no. For $m = \{0, 1\}$ we don't know the answer, and it remains an open question.

▶ **Claim 59.** *For every constant $\delta > 0$ and $t_1, t_2 \in \mathbb{N}, t_1 + t_2 \leq k$, there exist a constant $\beta > 0$ and a function $f : [N]^k \to [M]^k$ for $N, M \gg k$, such that $\alpha_{Z(t_1,t_2)}(f) = \epsilon \geq e^{-\delta k}$, but f is $(\epsilon^2, \frac{\beta}{\log k})$ far from any direct product function.*

Proof. Test 5 is symmetric with respect to t_1, t_2, so we can assume wlog that $t_1 \geq t_2$. We choose $N, M \geq e^{k^2}$ such that $M \leq \sqrt{N}$. We divide the proof into two cases, depending on t_1. For each of the two cases we construct a function with ℓ corrupted coordinates, such that $\alpha_{Z(t_1,t_2)}(f) = \epsilon \geq e^{-\delta k}$, and show that both these functions are $(\epsilon^2, \frac{\ell}{2k})$ far from direct product function.

1. Choose A, B, C to be a random partition of $[k]$, such that $|A| = t_1, |B| = t_2$.
2. Choose uniformly at random $x, y, z \in [N]^k$ such that $x_A = y_A$ and $y_B = z_B$.
3. Reject if $f(x)_A \neq f(y)_A$ or $f(z)_B \neq f(y)_B$, else accept.



Denote by $\alpha_{Z(t_1,t_2)}(f)$ the success probability of f on this test.

Test 5 "Z"-test with parameters t_1, t_2 (3-query test).

If $t_1 \leq 0.4k$

This case is similar to Example 3, and we provide here a detailed analysis. Let $f : [N]^k \to [M]^k$ be the constant function 1, i.e. $f(x) = 1, \ldots 1$ for every $x \in [N]^k$, but for every $x \in [N]^k$ we corrupt $f(x)$ on $\ell \leq \frac{1}{10}k$ random coordinates $i_x^{(1)}, \ldots i_x^{(\ell)}$ to random values in $[M] \setminus \{1\}$. The number of corrupted coordinates ℓ is decided later.

Let A, B, C, x, y, z the sets and inputs chosen in Test 5, since $t_2 \leq t_1 \leq 0.4k$, $|C| \geq 0.2k$. If all the corrupted coordinates of x, y, z are not in A and all the corrupted coordinates of y, z not in B, the output of f on all of the corrupted coordinates is not checked and the test passes.

$$\Pr\left[\text{Test passes}\right] \geq \Pr\left[i_x^{(1)}, \ldots i_x^{(\ell)} \notin A, i_y^{(1)}, \ldots i_y^{(\ell)} \notin A \cup B, i_z^{(1)}, \ldots i_z^{(\ell)} \notin B\right] \geq 0.1^{3\ell}.$$

The last inequality is because the corrupted coordinates on x, y, z are independent. For input x and $i_x^{(1)}, \ldots i_x^{(\ell)}$, even conditioning on $i_x^{(1)}, \ldots i_x^{(\ell-1)} \in C$, the probability of $i_x^{(\ell)}$ to be in C is at least 0.1 (since $\ell \leq 0.1k$), same for y, z.

We choose $\ell = \beta k$ for a constant β, such that $0.1^{3\ell} \geq e^{-\delta k}$, this means that f satisfies $\alpha_{Z(t_1,t_2)}(f) = \epsilon \geq e^{-\delta k}$.

We now show that f is $(\epsilon^2, \frac{\ell}{2k})$-far from every direct product function. We do it by describing a property of f, showing that our function satisfies it with high probability and that this property implies $(\epsilon^2, \frac{\ell}{2k})$-far from direct product function.

For every $i \in [k], b \in [N]$ let $G_{i,b}$ be

$$G_{i,b} = \left\{x \in [N]^k \;\middle|\; x_i = b\right\}.$$

The function f is called *balanced* if for every $i \in [k], b \in [N], a \in [M] \setminus \{1\}$,

$$\Pr_{x \in G_{i,b}}[f(x)_i = a] \leq \frac{2k}{M}.$$

We show that our random function f is balanced with probability almost 1. Fix $i \in [k], b \in [N], a \in [M] \setminus \{1\}$. By the definition of f, $\Pr_{x,\in G_{i,b}}[f(x)_i = a] \leq \frac{1}{M}$, and this is independent for each $x \in G_{i,b}$, therefore using Chernoff bound

$$\Pr\left[\sum_{x \in G_{i,b}} I(f(x)_i = a) \geq \frac{2}{M}N^{k-1}\right] \leq e^{-\frac{1}{3M}N^{k-1}}.$$

Preforming union bound over all $i \in [k], b \in [N], a \in [M]$, the probability that f is balanced is at least $1 - kNMe^{-\frac{1}{3M}N^{k-1}} \geq 1 - e^{-N}$.

Given that f is balanced and has exactly ℓ corrupted coordinates per input, we show it is $(\epsilon^2, \frac{\ell}{2k})$-far from direct product function. Let f be such function, and assume towards contradiction that there exist $g = (g_1, \ldots, g_k)$ that is $(\epsilon^2, \frac{\ell}{2k})$ close to f. Let $F = \left\{ x \in [N]^k \;\middle|\; f(x) \overset{\ell-1}{\approx} g(x) \right\}$, by our assumption $|F| \geq \epsilon^2 N^k$.

Let $F_{i,b} \subseteq G_{i,b}$ be the set

$$F_{i,b} = \{x \in F \mid \; x_i = b, g_i(x_i) = f(x)_i \neq 1\}.$$

Every $x \in F$ has ℓ coordinates $i \in [k]$ in which $f(x)_i \neq 1$. For every $x \in F$, $f(x) \overset{\ell-1}{\approx} g(x)$, so there must be $i \in [k]$ such that $f(x)_i = g_i(x_i) \neq 1$. Therefore, every $x \in F$ must be in at least one $F_{i,b}$, and the sets $\{F_{i,b}\}_{i\in[k],b\in[N]}$ must cover F, i.e. $F \subseteq \bigcup_{i\in[k],b\in[N]} F_{i,b}$.

By definition, all $x \in F_{i,b}$ satisfies $f(x)_i = g_i(b) \in [M] \setminus \{1\}$, since f is balanced, $|F_{i,b}| \leq \frac{2k}{M}|G_{i,b}| \leq \frac{2k}{M}N^{k-1}$.

$$|F| \leq \sum_{i\in[k],b\in[N]} |F_{i,b}| \leq Nk \cdot \frac{2k}{M}N^{k-1} \leq \frac{2k^2}{M}N^k \ll \epsilon^2 N^k$$

and we reached a contradiction to the assumption $|F| \geq \epsilon^2 N^k$.

If $t_1 > 0.4k$

In this case, we can't simply corrupt coordinates to random values, because it is possible that $t_1 + t_2 = t$, and all coordinates of $f(y)$ are checked. Instead, we corrupt coordinates in a more subtle way. We start by constructing a function $f : [N]^k \to [M]^k$ that has a single corrupted coordinate per input, and $\alpha_{Z(t_1,t_2)}(f) = \Omega(\frac{1}{k^2})$.

Let $f : [N]^k \to [M]^k$ be the constant 1 function (i.e. $f(x) = 1, \ldots 1$ for all x), and for every $b \in [N]$, let $p_b : [N] \to [M] \setminus \{1\}$ be a random function. For every input $x \in [N]^k$, we choose two random coordinates $i_x \neq j_x \in [k]$, i_x is the corrupted coordinate, and j_x is the master coordinate. We corrupt $f(x)$ by setting $b = x_{j_x}$ and

$$f(x)_{i_x} = p_b(x_{i_x}).$$

Let A, B, x, y, z be the sets and inputs chosen in the test, if $i_x = i_y$, $j_x = j_y$ and $i_x, j_x \in A$, then $f(x)_A = f(y)_A$ (because the corrupted coordinates are corrupted to the same value). If in addition $i_z \notin B$, then also $f(z)_B = f(y)_B$ (because y, z don't have any corrupted coordinates on B).

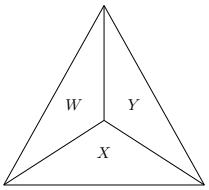
The probability of $i_x = i_y$ and $j_x = j_y$ is $\frac{1}{k^2}$, as they are both random indices in $[k]$. The probability of $i_x, j_x \in A, i_z \notin B$ is at least 0.3^3, therefore $\alpha_{Z(t_1,t_2)}(f) = \Omega(\frac{1}{k^2})$.

Instead of corrupting a single coordinate per input, we can corrupt $\ell \leq 0.1k$ different coordinates, by choosing different $i_x^{(1)}, \ldots, i_x^{(\ell)}$ and $j_x^{(1)}, \ldots, j_x^{(\ell)}$ for every $x \in [N]^k$, and continue as before. A similar probabilistic argument shows that that this function f has $\alpha_{Z(t_1,t_2)}(f) = \Omega(\frac{1}{k^{2\ell}})$ (conditioning on all other $i_x^{(1)}, \ldots i_x^{(\ell)}, j_x^{(1)}, \ldots, j_x^{(\ell-1)} \in A$, the probability of $j_x^{(\ell)} \in A$ is at least 0.2).

Fix a constant $\delta > 0$, in order of the function f to pass the test with probability $e^{-\delta k}$, the number of corrupted coordinates ℓ should satisfy $\frac{c}{k^{2\ell}} > e^{-\delta k}$, which means that we can choose $\ell = \beta\frac{k}{\log k}$ for some constant $\beta > 0$.

The constant function 1 is $(1, \frac{\ell}{k})$ close to f, we show that any direct product function $g = (g_1, \ldots, g_k)$ is $(\epsilon^2, \frac{\ell}{2k})$-far from f. Intuitively, it is true because the corrupted coordinates are corrupted to N different random functions, receiving values in M, for $k \ll N, M$. More

1. Choose disjoint $W, X, Y \subset [N]$ of size $\frac{k}{2}$.
2. Reject if $f(X \cup W)_W \neq f(Y \cup W)_W$, $f(X \cup Y)_Y \neq f(Y \cup W)_Y$ or $f(X \cup W)_X \neq f(X \cup Y)_X$, else accept.



Denote by $\alpha_{T_{set}}(f)$ the success probability of f on this test.

Test 6 Triangle test (3-query test, for even k).

formally, we show that with high probability the function f is also balanced, and use the proof of the previous case.

In the previous case, we showed that f is balanced with high probability by Chernoff bound over the inputs in $G_{i,b}$. This is not possible to do in our case, because for $x, y \in G_{i,b}$, there is a dependence between the values of the corrupted coordinates of x and y. Instead, we look at the random set of functions $\{p_b\}_{b\in[N]}$.

The function set $\{p_b\}_{b\in[N]}$ is called *balanced*, if for every $b' \in [N], a \in [M] \setminus \{1\}$, $\Pr_{b\in[N]}[p_b(b') = a] \leq 2\frac{1}{M}$.

Fix $b' \in [N], a \in [M] \setminus \{1\}$, a random function set $\{p_b\}_{b\in[N]}$ satisfies for every $b \in [N]$, $\Pr_{p_b}[p_b(b') = a] = \frac{1}{M-1}$, independently for each function p_b. Therefore using Chernoff bound,

$$\Pr_{\{p_b\}}\left[\sum_{b\in[N]} I(p_b(b') = a) > \frac{2N}{M}\right] \leq e^{-\frac{N}{4M}} \leq e^{-\frac{\sqrt{N}}{4}}.$$

Preforming union bound over all $b' \in [N], a \in [M] \setminus \{1\}$, a random function set $\{p_b\}_{b\in[N]}$ is balanced with probability at least $1 - NMe^{-\frac{\sqrt{N}}{4}}$.

We now show that a balanced function set $\{p_b\}_{b\in[N]}$ implies a balanced function f. Fix $i \in [k], b' \in [N], a \in [M]$, and let $A = \{b \in [N] \mid \ p_b(b') = a\}$, if $\{p_b\}_{b\in[N]}$ is balanced, then $|A| \leq \frac{2}{M}N$. The set $G_{i,b'}$ is a subcube of dimension $k-1$, so its coordinates are uniform in $[N]$, and by union bound

$$\Pr_{x\in G_{b',i}}[\exists j \in [k] \setminus \{i\} \text{ s.t } x_j \in A] \leq \frac{2k}{M}.$$

If there is no j such that $x_j \in A$, it is impossible that $f(x)_i = a$, because $f(x)_i$ is either 1, or $p_{x_j}(b')$ for some $j \in [k] \setminus \{i\}$. Therefore, at most $\frac{2k}{M}$ of $x \in G_{i,b'}$ can satisfy $f(x)_i = a$, and such f is balanced with probability 1.

The function f is balanced with ℓ corrupted coordinates per input, so by the previous case, f is $(\epsilon^2, \frac{\ell}{2k})$-far from direct product. ◀

Notice that in the proof, the range of $t_1 \leq 0.4k$ has a lower bound of $\ell = \beta k$, whereas in the second case, the lower bound is only $\ell = \frac{\beta k}{\log k}$.

The example in the proof can easily be transformed into a function on sets $f : \binom{[N]}{k} \to [M]^k$, which gives a bound on Test 4. This is done by choosing for each set $S \in \binom{[N]}{k}$ ℓ elements in S to corrupt and ℓ master elements (instead of coordinates) .

In Test 5 with $t_1 + t_2 = k$, we compare $f(y)$ on all coordinates, but only part of the coordinates of $f(x), f(z)$. What if we compare all coordinates of all three inputs? This brings us to the triangle test, ,Test 6, for functions over sets. In this test, every two out of the three inputs share a joint subset of size $\frac{k}{2}$, for this test we must assume that k is even.

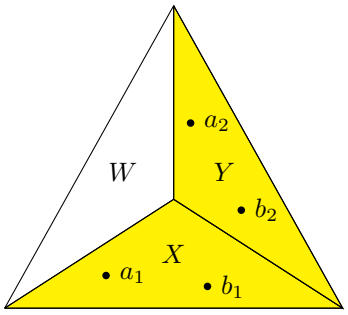


Figure 2 The set $S_2 = X \cup Y$ is marked in yellow.

▶ **Claim 60.** *For every constant $\delta > 0$, there exist a constant $\beta > 0$ and a function $f : \binom{[N]}{k} \to [M]^k$ with $N, M \gg k$, such that $\alpha_{T_{set}}(f) = \epsilon > e^{-\delta k}$, and f is $(\epsilon^2, \frac{\beta}{\log k})$ far from direct product function.*

Proof. The function f that we describe in this proof is similar to the function from the previous proof, we only need to modify it slightly such that there is the same number of corrupted elements in each half of the inputs. We start by describing a function with two corrupted elements per input.

Let $f : \binom{[N]}{k} \to [M]^k$ be the constant function 1, i.e. $f(S) = 1, \ldots 1$ for every set S, and for every $b \in [N]$ we choose a random function $p_b : [N] \to [M] \setminus \{1\}$. For every $S \in \binom{[N]}{k}$, we choose two elements to corrupt $a_1, a_2 \in S$ and two master elements $b_1, b_2 \in S$. Then, we set $f(S)_{a_1} = p_{b_1}(a_1)$ and $f(S)_{a_2} = p_{b_2}(a_2)$.

Suppose W, X, Y are the sets chosen in Test 6, fix $a_1, b_1 \in X, a_2, b_2 \in Y$ and $a_3, b_3 \in W$. If the following three events hold, the test passes, see Figure 2.

1. In the set $S_2 = X \cup Y$ the elements chosen to corrupt are a_1, a_2 with the master elements b_1, b_2 respectively.
2. In the set $S_1 = X \cup W$ the elements chosen to corrupt are a_1, a_3 with the master elements $b_1\, b_3$ respectively.
3. In the set $S_3 = Y \cup W$ the elements chosen to corrupt are a_2, a_3 with the master elements b_2, b_3 respectively.

If the three events hold, then on every check of the test, both the corrupted element and its master element are the same in both inputs, so they are corrupted to the same value and the check passes.

The probability of each event is at least $\frac{1}{k^4}$, and the event are independent, since the choice of which elements to corrupt is done independently for each $S \in \binom{[N]}{k}$. Therefore the function f passes Test 6 with probability at least $\frac{1}{k^{12}}$. It is possible to do a more careful analysis and get a higher success probability bound, but it is not important in our case.

If we corrupt 2ℓ elements per set $S \in \binom{[N]}{k}$, similar analysis shows that f satisfies $\alpha_{T_{set}}(f) = \Omega(\frac{1}{k^{12\ell}})$. Setting $\ell = \frac{\beta k}{\log k}$ for some constant β, we get f such that $\alpha_{T_{set}}(f) \geq e^{-\delta k}$.

We show that f with 2ℓ corrupted coordinates is $(\epsilon^2, \frac{\ell}{2k})$-far from direct product function in a very similar way to the previous proof. As we have seen in the proof of Claim 59, the random function set $\{p_b\}_{b \in [N]}$ is balanced with probability at least $1 - MNe^{-\frac{1}{4}\sqrt{N}}$.

For every $b \in [N]$, let $G_b = \{S \subset [N] \,|\, |S| = k, b \in S\}$, we say that the function $f : \binom{[N]}{k} \to [M]^k$ is *balanced* if for every $b' \in [N], a \in [M] \setminus \{1\}$,

$$\Pr_{S \in G_{b'}}[f(S)_{b'} = a] \leq \frac{2k}{M}.$$

We show that every f with a balanced function set $\{p_b\}_{b\in[N]}$ is balanced. Fix $b' \in [N], a \in [M]$, and let $A = \{b \in [N] \mid \ p_b(b') = a\}$, for a balanced function set, $|A| \leq \frac{2M}{N}$. Like previously, the set $G_{b'}$ is actually equivalent to all subset of size $k-1$ of elements in $[N]\setminus\{b'\}$, therefore a uniform $S \in G_{b'}$ contains $b \in A$ with probability at most $\frac{2k}{M}$, and f is balanced.

Assume towards contradiction that f is $(\epsilon^2, \frac{\ell}{2k})$ close to a direct product function $g : [N] \to [M]$, and let F be set set of inputs in which $f(S) \overset{\ell-1}{\approx} g(S)$. Similar to the previous proof, for every $b \in [N]$ let $F_b = \{S \in F \mid \ b \in S, f(S)_b = g(b) \neq 1\}$.

Since g approximated F up to $\ell-1$ elements, and f has ℓ corrupted elements, every $S \in F$ is in some $F_{b'}$, and $F \subseteq \cup_{b'\in[N]} F_{b'}$. Since f is balanced, for every $b' \in [N]$, $|F_{b'}| \leq \frac{2k}{M}|G_{b'}|$,

$$|F| \leq \sum_{b'\in[N]} |F_{b'}| \leq N\frac{2k}{M}|G_{b'}| \leq \frac{2k^2}{M}\left|\binom{[N]}{k}\right|.$$

The last inequality, is because each $S \in \binom{[N]}{k}$ is in at most k sets $G_{b'}$. This is a contradiction of $|F| \geq \epsilon^2\left|\binom{[N]}{k}\right|$. ◀

References

1. Sanjeev Arora and Madhu Sudan. Improved low-degree testing and its applications. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 485–495. ACM, 1997.
2. Vitaly Bergelson, Terence Tao, and Tamar Ziegler. An inverse theorem for the uniformity seminorms associated with the action of $\mathbb{F}_p^\infty$. *Geometric and Functional Analysis*, 19(6):1539–1596, 2010.
3. Amey Bhangale, Irit Dinur, and Inbal Livni Navon. Cube vs. cube low degree test. In *Proceedings of the 2017 Conference on Innovations in Theoretical Computer Science, ITCS*, 2017.
4. Irit Dinur. The PCP theorem by gap amplification. *Journal of the ACM (JACM)*, 54(3):12, 2007.
5. Irit Dinur and Elazar Goldenberg. Locally testing direct product in the low error range. In *Foundations of Computer Science, 2008. FOCS'08. IEEE 49th Annual IEEE Symposium on*, pages 613–622. IEEE, 2008.
6. Irit Dinur, Subhash Khot, Guy Kindler, Dor Minzer, and Muli Safra. Towards a proof of the 2-to-1 games conjecture? *Electronic Colloquium on Computational Complexity (ECCC)*, 2016. URL: `https://eccc.weizmann.ac.il/report/2016/198/`.
7. Irit Dinur and David Steurer. Direct product testing. In *Computational Complexity (CCC), 2014 IEEE 29th Conference on*, pages 188–196. IEEE, 2014.
8. Oded Goldreich and Shmuel Safra. A combinatorial consistency lemma with application to proving the PCP theorem. *SIAM Journal on Computing*, 29(4):1132–1154, 2000.
9. Russell Impagliazzo, Valentine Kabanets, and Avi Wigderson. New direct-product testers and 2-query PCPs. *SIAM Journal on Computing*, 41(6):1722–1768, 2012.
10. Subhash Khot, Dor Minzer, and Muli Safra. On Independent Sets, 2-to-2 Games and Grassmann Graphs. *Electronic Colloquium on Computational Complexity (ECCC)*, 2016.
11. Elchanan Mossel, Krzysztof Oleszkiewicz, and Arnab Sen. On reverse hypercontractivity. *Geometric and Functional Analysis*, 23(3):1062–1097, 2013.
12. Ran Raz. A parallel repetition theorem. *SIAM Journal on Computing*, 27(3):763–803, 1998.
13. Ran Raz and Shmuel Safra. A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 475–484. ACM, 1997.

A Chernoff and Hypercontractivity Proofs

Proof of Claim 11. For each element $i \in D$, we define the indicator random variable I_i to indicate that $i \in A$. In this notation

$$|A \cap D| = \sum_{i \in D} I_i.$$

We want to use Chernoff bound on I_i, but since A is of fixed size, the indicator variables are not independent. Instead, we define for each i the new random variables J_i that are independent.

For (2), let

$$J_i = \begin{cases} 1 & \text{w.p } \frac{3}{20} \\ 0 & \text{w.p } 1 - \frac{3}{20} \end{cases}.$$

For every $i \in D$ and every fixed value $b \in \{0,1\}^{|D|}$ of the indicators $\{I_l, l \neq i\}$, $\Pr[I_i = 1 \mid \forall l \neq i, I_l = b_l] \leq \Pr[J_i = 1]$. In the worse case, they are all set to 0 (none is in A), and $\Pr[I_i = 1] = \frac{3}{20}$. Therefore, we can use Chernoff bound on the random variables J_i and get a result for I_i:

$$\Pr_A\left[\sum_{i \in D} I_i \geq \frac{1}{5}|D|\right] \leq \Pr_J\left[\sum_{i \in D} J_i \geq \frac{1}{5}|D|\right] \leq e^{-\frac{1}{320}|D|}$$

For (3), we define

$$J_i = \begin{cases} 1 & \text{w.p } \frac{1}{15} \\ 0 & \text{w.p } 1 - \frac{1}{15} \end{cases}.$$

In this case, for every $i \in D$ and fixed value $b \in \{0,1\}^{|D|}$, $\Pr[I_i = 1 \mid \forall l \neq i, I_l = b_l] \geq \Pr[J_i = 1]$, and

$$\Pr_A\left[\sum_{i \in D} I_i \leq \frac{1}{20}|D|\right] \leq \Pr_J\left[\sum_{i \in D} J_i \leq \frac{1}{20}|D|\right] \leq e^{-\frac{1}{60}|D|}. \quad \blacktriangleleft$$

Proof of Corollary 15. $|A| \geq |B|$ implies $a \leq b$, we know that

$$e^{-\frac{\rho a b}{2(1-\rho)}} \geq e^{-\frac{\rho b^2}{2(1-\rho)}} = \Pr_{x \in [N]^k}[x \in B]^{\frac{\rho}{1-\rho}}.$$

Similarly

$$e^{-\frac{(2-\rho)(a^2+b^2)}{4(1-\rho)}} = e^{\frac{2-\rho}{2(1-\rho)} \cdot \left(-\frac{a^2}{2} - \frac{b^2}{2}\right)} = e^{\left(1+\frac{\rho}{2(1-\rho)}\right) \cdot \left(-\frac{a^2}{2} - \frac{b^2}{2}\right)} =$$
$$\Pr_{x \in [N]^k}[x \in B]^{1+\frac{\rho}{2(1-\rho)}} \Pr_{x \in [N]^k}[x \in A]^{1+\frac{\rho}{2(1-\rho)}}$$

Together we get

$$\Pr_{x,y}[x \in A, y \in B] \geq \Pr_{x \in [N]^k}[x \in A]^{1+\frac{\rho}{2(1-\rho)}} \Pr_{x \in [N]^k}[x \in B]^{1+\frac{3\rho}{2(1-\rho)}}. \quad \blacktriangleleft$$

Proof of Claim 16. We notice that regardless which of the sets G, L is the largest, by Corollary 15,

$$\Pr_{w\in[N]^k,(v,J)\in\mathcal{N}_{\frac{3}{4}}(w)}[w\in L, v\in G] \geq \left(\Pr_w[w\in L]\right)^{\frac{11}{2}} \nu^{\frac{11}{2}}.$$

By the definition of L,

$$\Pr_{w\in[N]^k,(v,J)\in\mathcal{N}_{\rho}(w)}[w\in L, v\in G] \leq \Pr_w[w\in L]\eta.$$

Therefore

$$\Pr_w[w\in L]^{\frac{9}{2}} \leq \nu^{-\frac{11}{2}}\eta.$$ ◀

B Tuples to Sets Local Structure Proof

In this section we prove Lemma 37, restricted global structure for sets, we restate it bellow.

▶ **Lemma 37.** *There exist a small constants $\delta > 0$, such that for every constant $\lambda > 0$ and large enough $k \in \mathbb{N}$ such that $N > k^2 e^{10\delta\lambda k}$, the following holds,*

For every function $f : \binom{[N]}{k} \to [M]^k$, if $\alpha_{Z_{set}(\frac{k}{10})}(f) = \epsilon > e^{-\delta\lambda k}$, then at least $(1-\epsilon^2-\frac{k^2}{N})$ of the good pairs $W \in \binom{[N]}{\frac{k}{10}}, X \in \binom{[N]}{\frac{9k}{10}}$ are DP pairs, i.e. there exist $g_{X,W} : [N] \to [M]$ such that

$$\Pr_Y\left[f(Y\cup W)_Y \overset{3\alpha k}{\not\approx} g_{X,W}(Y) \;\middle|\; Y\cap W=\emptyset, f(X\cup W)_W = f(Y\cup W)_W\right] \leq 2\epsilon^2.$$

In order to prove the lemma, for every function $f : \binom{[N]}{k} \to [M]^k$ we define a function $f' : [N]^k \to [M]^k \cup \perp$. For every $S \subset [N]$, we assume that the output of $f(S)$ is ordered in an ascending order over the elements or S.

In order to simplify the notation, for every string $x \in [N]^k$, we define $U(x) = 1$ if x has unique coordinates, i.e there is no $i \neq j$ such that $x_i = x_j$, else $U(x) = 0$.

▶ **Definition 61.** Given a function $f : \binom{[N]}{k} \to [M]^k$, let $f' : [N]^k \to [M]^k \cup \perp$ be defined as follows. For every $x \in [N]^k$ let X be the set of elements in x,

$$f'(x) = \begin{cases} \pi(f(X)) & U(x) = 1 \\ \perp & U(x) = 0 \end{cases}.$$

Where $\pi \in \mathcal{S}_k$ is the permutation from the ascending order over the elements of X to x.

For a set $S \subset [N]$ of size k and a permutation $\pi \in \mathcal{S}_k$, we denote by $\pi(S) \in [N]^k$ the string generated by applying π on the elements of S ordered in an ascending order. Therefore, for every $X \in \binom{[N]}{k}$, $f'(\pi(X)) = \pi(f(X))$.

▶ **Definition 62.** Let $\mathcal{D} : \binom{[N]}{\frac{k}{10}} \times \binom{[N]}{\frac{9k}{10}} \times \binom{[N]}{\frac{9k}{10}} \to [0,1]$ be the following distribution:

1. Choose $W \subset [N]$ of size $\frac{k}{10}$.
2. Choose $X \subset [N]$ of size $\frac{9k}{10}$ such that $X \cap W = \emptyset$.
3. Choose $Y \subset [N]$ of size $\frac{9k}{10}$ such that $Y \cap W = \emptyset$.

Let $\mathcal{D}' : \binom{[k]}{\frac{k}{10}} \times [N]^k \times [N]^k \to [0,1]$ be the following distribution:

1. Choose a set $A \subset [k]$ of size $\frac{k}{10}$.

2. Choose $x \in [N]^k$ such that $U(x) = 1$.
3. Choose $y \in [N]^k$ such that $x_A = y_A$ and $U(y) = 1$.

Fixing a set $A \subset [k]$ and $x \in [N]^k$ such that $U(x) = 1$, we denote by $\mathcal{D}'|A, x$ the distribution over y, conditioning on A, x being already chosen. Similarly for $W, X \subset [N]$, we define $\mathcal{D}|W, X$ the distribution over Y.

We can easily see that if we pick $(W, X, Y) \sim \mathcal{D}$, then choose a random set A and random permutations $\pi_1 \in \mathcal{S}_{\frac{k}{10}}, \pi_2, \pi_3 \in \mathcal{S}_{\frac{k}{10}}$, and set $x = (\pi_1(W)_A, \pi_2(X)_{\bar{A}})$,$y = (\pi_1(W)_A, \pi_3(Y)_{\bar{A}})$, we get $(A, x, y) \sim \mathcal{D}'$.

For each two sets W, X, let $x = (\pi_1(W)_A, \pi_2(X)_{\bar{A}})$ for an arbitrary $A \subset [k]$ and π_1, π_2, then the distribution $y \sim \mathcal{D}'|A, x$ is the same distribution as $(\pi_1(W)_A, \pi_3(Y)_{\bar{A}})$ for $Y \sim \mathcal{D}|W, X$ and uniform $\pi_3 \in \mathcal{S}_{\frac{9k}{10}}$.

We further notice that the distribution $(W, X, Y) \sim \mathcal{D}$ is the distribution used in Test 4. The distribution $(A, x, y) \sim \mathcal{D}'$ is the distribution of Test 2 with $t = \frac{k}{10}$ conditioning on $U(x) = U(y) = 1$.

Let $p_1 = \Pr_{x \in [N]^k}[U(x) = 0]$. For every $x \in [N]^k$ such that $U(x) = 1$ and a set $A \subset [k]$, let $p_2 = \Pr_y[U(y) = 0 \mid y_A = x_A]$ (p_2 is the same for every A, x such that $U(x) = 1$). We bound the probabilities p_1, p_2.

Choosing a uniform $x \in [N]^k$ can be done coordinate by coordinate. For each coordinate i, the probability that $x_i = x_j$ for $j < i$ is less than $\frac{i-1}{N}$, therefore

$$p_1 = \Pr_{x \in [N]^k}[U(x) = 0] \leq \sum_{i=1}^{k} \frac{i-1}{N} \leq \frac{k^2}{2N}.$$

Similarly, we can think of picking y given A, x as starting with the fixed y_A (which doesn't contain two identical coordinates as $U(x) = 1$) and choosing coordinates one by one.

$$p_2 = \Pr_y[U(y) = 0 \mid y_A = x_A] \leq \sum_{i=\frac{k}{10}}^{k} \frac{i-1}{N} \leq \frac{k^2}{2N}.$$

▶ **Claim 63.** *For every function $f : \binom{[N]}{k} \to [M]^k$, the function $f' : [N]^k \to [M]^k$ from Definition 61 satisfies*

$$\alpha_{V(\frac{k}{10})}(f') = (1 - p_1)(1 - p_2) \Pr[f \text{ passes } Item3 \text{ of } Test\ 4].$$

Proof. Fix a function $f : \binom{[N]}{k} \to [M]^k$, and let $f' : [N]^k \to [M]^k$ be the function from Definition 61.

If either $U(x) = 0$ or $U(y) = 0$, by definition f' outputs $\perp$ and the test fails. If we condition on $U(x) = U(y) = 1$, the test distribution equals $\mathcal{D}'$. Let W be the set of elements of x_A, X of $x_{\bar{A}}$ and Y of $y_{\bar{A}}$, then $(W, X, Y) \sim \mathcal{D}$.

For every A, x, y such that $U(x) = U(y) = 1$ and $x_A = y_A$, the permutation $\pi_1 \in \mathcal{S}_{\frac{k}{10}}$ from the ascending order in W to the order of x_A satisfies $f'(x)_A = \pi_1(f(X, W)_W)$, and $f'(y)_A = \pi_1(f(Y, W)_W)$. Therefore, $f'(x)_A = f'(y)_A \iff f(X, W)_W = f(Y, W)_W$.

This implies that

$$\begin{aligned}
\Pr[f' \text{ passes } Test\ 2] &= \Pr_{A,x,y}[f'(x)_A = f'(y)_A \mid x_A = y_A] \\
&= \Pr_{A,x,y}[U(x) = U(y) = 1 \mid x_A = y_A] \Pr_{(A,x,y) \sim \mathcal{D}'}[f'(x)_A = f'(y)_A] \\
&= (1 - p_1)(1 - p_2) \Pr_{(W,X,Y) \sim \mathcal{D}}[f(X, W)_W = f(Y, W)_W] \\
&= (1 - p_1)(1 - p_2) \Pr[f \text{ passes } Item3 \text{ of } Test\ 4].
\end{aligned}$$

Where $\Pr_{A,x,y}[U(x)=U(y)=1 \mid x_A=y_A]=(1-p_1)(1-p_2)$ by the definition of p_1,p_2. ◀

▶ **Claim 64.** *For every function on sets $f:\binom{[N]}{k}\to[M]^k$, the function $f':[N]^k\to[M]^k$ from Definition 61 satisfies the following. For every disjoint $W\in\binom{[N]}{\frac{k}{10}}, X\in\binom{[N]}{\frac{9k}{10}}$, every set $A\subset[k], |A|=\frac{k}{10}$ and every permutations $\pi_1\in\mathcal{S}_{\frac{k}{10}},\pi_2\in\mathcal{S}_{\frac{9k}{10}}$, the pair $(A,x=(\pi_1(W)_A,\pi_2(X)_{\bar{A}}))$ satisfies*

$$\Pr_y[f'(x)_A=f'(y)_A \mid y_A=x_A]=(1-p_2)\Pr_{Y\sim\mathcal{D}|W,X}[f(X\cup W)_W=f(Y\cup W)_W].$$

Proof. Fix a function $f:\binom{[N]}{k}\to[M]^k$, and let $f':[N]^k\to[M]^k$ be the function from Definition 61. Fix two disjoint subsets $W\in\binom{[N]}{\frac{k}{10}}, X\in\binom{[N]}{\frac{9k}{10}}$, a subset $A\subset[k], |A|=\frac{k}{10}$, and permutations $\pi_1\in\mathcal{S}_{\frac{k}{10}},\pi_2\in\mathcal{S}_{\frac{9k}{10}}$. Set $x=(\pi_1(W)_A,\pi_2(X)_{\bar{A}})$, since X,W are disjoint, $U(x)=1$. By the definition of f', $f'(x)_A=\pi_1(f(X,W)_W)$.

Let $y\in[N]^k$ be a random string such that $x_A=y_A$, if $U(y)=0$, then $f'(y)=\perp$ and $f'(x)_A\neq f'(y)_A$. By definition, $p_2=\Pr_y[U(y)=0|x_A=y_A]$. If we condition on $U(y)=1$, the distribution over y is $\mathcal{D}'|A,x$. If we take Y to be the elements of $y_{\bar{A}}$, then the distribution over Y is $\mathcal{D}|W,X$.

For y such that $U(y)=1$, by the definition of f', $f'(y)_A=\pi_1(f(Y,W)_W)$, and therefore $f'(x)_A=f'(y)_A \iff f(X,W)_W=f(Y,W)_W$.

$$\begin{aligned}\Pr_y[f'(x)_A=f'(y)_A \mid y_A=x_A]&=\Pr_y[U(y)=0 \mid x_A=y_A]\Pr_{y\sim\mathcal{D}'|A,x}[f'(x)_A=f'(y)_A]\\&=(1-p_2)\Pr_{Y\sim\mathcal{D}|W,X}[f(X\cup W)_W=f(Y\cup W)_W].\end{aligned}$$ ◀

Proof of Lemma 37. Let $f:\binom{[N]}{k}\to[M]^k$ be the function such that $\alpha_{Z_{set}(\frac{k}{10})}(f)=\epsilon>e^{-\delta\lambda k}$, and let $f':[N]^k\to[M]^k$ be the function from Definition 61. By Claim 63, f' passes Test 2 with probability $\epsilon'=(1-p_1)(1-p_2)\epsilon$, therefore, Theorem 21 holds for the function f'.

By Claim 64, for every disjoint $W\in\binom{[N]}{\frac{k}{10}}, X\in\binom{[N]}{\frac{9k}{10}}$,

$$\Pr_y[f'(x)_A=f'(y)_A \mid y_A=x_A]=(1-p_2)\Pr_{Y\sim\mathcal{D}_{W,X}}[f(X\cup W)_W=f(Y\cup W)_W].$$

Setting $\eta=1-p_1$, this means that if X,W satisfies $\Pr_Y[f(X\cup W)_W=f(Y\cup W)_W]\geq\eta\frac{\epsilon}{2}$, then for every set $A\subset[k]$ and permutations π_1,π_2, the pair $(A,x=(\pi_1(W)_A,\pi_2(X)_{\bar{A}}))$ satisfies $\Pr_y[f'(x)_A=f'(y)_A \mid y_A=x_A]\geq\frac{\epsilon'}{2}$.

Theorem 21 implies that with probability $1-\epsilon'^2$ a good $\tau\sim\mathcal{D}$ (equivalent to A,x that satisfies $\Pr_y[f'(x)_A=f'(y)_A \mid y_A=x_A]\geq\frac{\epsilon'}{2}$) is a DP-restriction. Since every W,X corresponds for the same number of (A,x), for at least $(1-\epsilon'^2)\geq(1-\epsilon^2-\frac{k^2}{N})$ of the sets W,X, there exist at least one set A and permutations π_1,π_2 such that $\tau=(A,x,f'(x)_A)$ is a DP restriction, for $x=(\pi_1(W)_A,\pi_2(X)_{\bar{A}})$.

Let W,X be such sets, i.e. there exist $A\subset[k]$ and permutations π_1,π_2 such that $\tau=(A,x,f'(x)_A)$ is a DP-restriction, for $x=(\pi_1(W)_A,\pi_2(X)_{\bar{A}})$. We show that (W,X) are a DP-pair. Let $g^\tau=g_1^\tau,\ldots g_{\frac{9k}{10}}^\tau, g_i^\tau:[N]\to[M]$ be the direct product function of τ. We define $g_{W,X}:[N]\to[M]$ to be the following function, for every $a\in[N]$, $g_{W,X}(a)$ is the most frequent value $g_i^\tau(a)$, among all $i\in\frac{9k}{10}$.

We recall that $\mathcal{V}_\tau=\left\{w\in[N]^{\bar{A}}\;\middle|\; f'(x_A,w)_A=f'(x)_A\right\}$ and denote by $\mathcal{V}_{W,X}$ the analog in sets,

$$\mathcal{V}_{W,X}=\left\{Y\in\binom{[N]}{\frac{9k}{10}}\;\middle|\; Y\cap W=\emptyset, f(Y,W)_W=f(X,W)_W\right\}.$$

We notice that for every $w \in \mathcal{V}_\tau$, $f'(x_A, w) \neq \perp$, so it has unique coordinates, $U(x_A, w) = 1$.

In these notations, Theorem 21 implies $\Pr_{w \in \mathcal{V}_\tau}\left[f'(x_A, w)_{\bar{A}} \overset{\alpha k}{\not\approx} g^\tau(w)\right] \leq \epsilon'^2$, and we need to prove the analog statement for $Y \in \mathcal{V}_{W,X}$.

We describe the following random process: for every $Y \in \mathcal{V}_{W,X}$, we choose a random permutation π_3 and set $w = \pi_3(Y)$. We notice that for every $Y \in \mathcal{V}_{W,X}$, $f(Y,W)_W = f(X,W)_W$, and by the definition of f' this implies that $f'(x_A, w)_A = f'(x)_A$, so $w \in \mathcal{V}_\tau$. Moreover, for every $w \in \mathcal{V}_\tau$ exists exactly one $Y \in \mathcal{V}_{W,X}$ and permutation π_3 such that $w = \pi_3(Y)$.

Suppose $Y \in \mathcal{V}_{W,X}$ such that $g_{W,X}(Y) \overset{3\alpha k}{\not\approx} f(Y \cup W)_Y$, and let $B \subset Y$ be the set of elements that $g_{W,X}(Y), f(Y \cup W)_Y$ differ on, i.e. for every $b \in B$, $g_{W,X}(b) \neq f(Y \cup W)_b$. Since $g_{W,X}$ is the most frequent value among $g_i^\tau(b)$, for at least half of the locations i, $g_i^\tau(b) \neq f(Y \cup W)_b$.

For a random permutation π_3, each $b \in B$ has probability of at least $\frac{1}{2}$ to fall into a "bad location", i.e i such that $g_i^\tau(b) \neq f(Y \cup W)_b$. Since α is a very small constant, even conditioning on αk of $b \in B$ to be in a bad location, the probability of $b' \in B$ to fall into a bad location is at least $\frac{2}{5}$. By Chernoff bound, with probability larger than $1 - e^{-\frac{1}{100}\alpha k}$, π_3 is such that at least $\frac{1}{3}$ of $b \in B$ are in a "bad location". By the definition of f', this implies that $f'(x_A, w)_{\bar{A}} \overset{\alpha k}{\not\approx} g^\tau(w)$.

Therefore, we get that

$$\Pr_{Y \in \mathcal{V}_{W,X}}\left[g_{W,X}(Y) \overset{3\alpha k}{\not\approx} f(Y \cup W)_Y\right]\left(1 - e^{-\frac{1}{100}\alpha k}\right) \leq \Pr_{w \in \mathcal{V}_\tau}\left[f'(x_A, w)_{\bar{A}} \overset{\alpha k}{\not\approx} g^\tau(w)\right] \leq \epsilon'^2.$$

Which implies that

$$\Pr_{Y \in \mathcal{V}_{W,X}}\left[g_{W,X}(Y) \overset{3\alpha k}{\not\approx} f(Y \cup W)_Y\right] \leq \epsilon'^2 + e^{-\frac{1}{100}\alpha k} \leq 2\epsilon^2. \quad \blacktriangleleft$$

On the Polynomial Parity Argument Complexity of the Combinatorial Nullstellensatz*

Aleksandrs Belovs[1], Gábor Ivanyos[2], Youming Qiao[3], Miklos Santha[4], and Siyi Yang[5]

1 **Faculty of Computing, University of Latvia, Riga, Lettland**
stiboh@gmail.com

2 **Institute for Computer Science and Control, Hungarian Academy of Sciences, Budapest, Hungary**
Gabor.Ivanyos@sztaki.mta.hu

3 **Centre for Quantum Software and Information, University of Technology Sydney, Sydney, Australia**
jimmyqiao86@gmail.com

4 **IRIF, Université Paris Diderot, CNRS, Paris, France; and Centre for Quantum Technologies, National University of Singapore and MajuLab, CNRS, Singapore**
santha@irif.fr

5 **Centre for Quantum Technologies, National University of Singapore, Singapore**
ysyshtc@gmail.com

Abstract

The complexity class PPA consists of NP-search problems which are reducible to the parity principle in undirected graphs. It contains a wide variety of interesting problems from graph theory, combinatorics, algebra and number theory, but only a few of these are known to be complete in the class. Before this work, the known complete problems were all discretizations or combinatorial analogues of topological fixed point theorems.

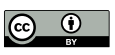
Here we prove the PPA-completeness of two problems of radically different style. They are PPA-Circuit CNSS and PPA-Circuit Chevalley, related respectively to the Combinatorial Nullstellensatz and to the Chevalley-Warning Theorem over the two elements field $\mathbb{F}_2$. The input of these problems contain PPA-*circuits* which are arithmetic circuits with special symmetric properties that assure that the polynomials computed by them have always an even number of zeros. In the proof of the result we relate the multilinear degree of the polynomials to the parity of the *maximal parse subcircuits* that compute monomials with maximal multilinear degree, and we show that the maximal parse subcircuits of a PPA-circuit can be paired in polynomial time.

1998 ACM Subject Classification F.1.3 Complexity Measures and Classes

Keywords and phrases Chevalley-Warning theorem, Combinatorial Nullstellensatz, PPA

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.30

* This research was partially funded by the Singapore Ministry of Education and the National Research Foundation, also through the Tier 3 Grant "Random numbers from quantum processes," MOE2012-T3-1-009. The research was also supported by the ERC Advanced Grant MQC, the French ANR Blanc program under contract ANR-12-BS02-005 (RDAM project), the Hungarian National Research, Development and Innovation Office – NKFIH Grant K115288 and the Australian Research Council DECRA DE150100720.



32nd Computational Complexity Conference (CCC 2017).
Editor: Ryan O'Donnell; Article No. 30; pp. 30:1–30:24

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

1 Introduction

1.1 The class PPA

The complexity class TFNP [21] consists of NP-search problems corresponding to total relations. In the last 25 years various subclasses of TFNP have been thoroughly investigated. The polynomial parity argument classes PPA and PPAD were defined in the seminal work of Papadimitriou [22]. PPA consists of the search problems which are reducible to the parity principle stating that in an undirected graph the number of odd vertices is even. The more restricted class PPAD is based on the analogous principle for directed graphs.

The class PPAD contains a relatively large number of complete problems from various areas of mathematics. In his paper Papadimitrou [22] has already shown that among others the 3-dimensional Sperner and Brouwer problems, as well as the Exchange Equilibrium problem from mathematical economics were PPAD-complete. A few years later Chen and Deng [9] proved that 2-dimensional Sperner was also PPAD-complete, and after a sequence of beautiful papers Chen and Deng [10] has established the PPAD-completeness of computing 2-player Nash equilibrium, see also [11]. Kintali [18] has compiled a list of 25 PPAD-complete problems; the list is far from complete.

In comparison with PPAD, relatively few complete problems are known in the class PPA, all of which are discretizations or combinatorial analogues of topological fixed point theorems. While the original paper of Papadimitriou [22] exhibited a large collection of problems in PPA, none of them was proven to be PPA-complete. Historically the first PPA-completeness result was given by Grigni [14] who, realizing that analogues of PPAD-complete problems in non-orientable spaces could become PPA-complete, has shown the PPA-completeness of the Sperner problem for a non-orientable 3-dimensional space. This result was strengthened by Friedl et al. [17] to a non-orientable and locally 2-dimensional space. Up to our knowledge, until 2015 just these two problems were known to be PPA-complete. Last year Deng et al. [13] established the PPA-completeness of several 2-dimensional problems on the Möbius band, including Sperner and Tucker, and they have obtained similar results for the Klein bottle and the projective plane. Recently Aisenberg, Bonet and Buss [1] have shown that 2-dimensional Tucker in the Euclidean space was PPA-complete.

Compared to the fundamental similarity of these complete problems in PPA, the list of problems in the class for which no completeness result is known is very rich. Already in Papadimitriou's paper [22] we find problems from graph theory, such as Smith and Hamiltonian decomposition, from combinatorics, such as Necklace splitting and Discrete Ham sandwich (the proof in [23] that these problems are in PPAD was incorrect [1]), and from algebra, a variant of Chevalley's theorem over the 2 elements field $\mathbb{F}_2$, which we call Explicit Chevalley. Cameron and Edmonds [8] gave new proofs based on the parity principle for a long series of theorems from graph theory [25, 29, 6, 5, 7], the corresponding search problems are therefore in PPA. Recently Jeřábek [15] has put several number theoretic problems, such as square root computation and finding quadratic nonresidues modulo n into PPA, and he has also shown that Factoring is in PPA under randomized reduction.

1.2 Our contribution

The main result of this paper is that two appropriately defined problems related to Chevalley-Warning Theorem [12, 28] and to Alon's Combinatorial Nullstellensatz [2] over $\mathbb{F}_2$ are complete in PPA. These are the first PPA-completeness results involving problems which are not inspired by topological fixed point theorems.

The Chevalley-Warning Theorem is a classical result about zeros of polynomials. It says that if $P_1, \ldots, P_k$ are n-variate polynomials over a field of characteristic p such that the sum of their degrees is less than n, then the number of common zeros is divisible by p. The Combinatorial Nullstellensatz (CNSS) of Alon states that if P is an n-variate polynomial over $\mathbb{F}$ whose degree is $d_1 + \cdots + d_n$, and this is certified by the monomial $cx_1^{d_1} \cdots x_n^{d_n}$, for some $c \neq 0$, then in $S_1 \times \cdots \times S_n \subseteq \mathbb{F}^n$ there exists a point where P is not zero, whenever $|S_i| > d_i$, for $i = 1, \ldots, n$. The CNSS has found a wide range of applications among others in graph theory, combinatorics and additive number theory [2, 3].

Over the field $\mathbb{F}_2$ the two theorems greatly simplify via the notion of *multilinear degree*. For any polynomial P over $\mathbb{F}_2$, there exists a unique multilinear polynomial M such that P and M compute the same function on $\mathbb{F}_2^n$. We call the degree of M the multilinear degree of P, denoted as $\mathrm{mdeg}(P)$. We use $\deg(P)$ to denote the usual degree of P. Then the Chevalley-Warning Theorem and the CNSS over $\mathbb{F}_2$ are equivalent to the following statement: *An n-variate $\mathbb{F}_2$-polynomial has an odd number of zeros if and only if its multilinear degree is n.* The natural search problem corresponding to the CNSS therefore is: given an n-variate polynomial P whose multilinear degree is n, find a point a where $P(a) = 1$. Similarly, the search problem corresponding to the Chevalley-Warning Theorem is: given an n-variate polynomial P whose multilinear degree is less than n and a zero of P, find another zero.

Obviously, these problems are not yet well defined algorithmically, since it is not specified, how the polynomial P is given. The starting point of our investigations is the result of Papadimitriou about some instantiation of the Chevalley-Warning Theorem. Specifically, in [22] Papadimitriou considered the following problem. Let the polynomials $P_1, \ldots, P_k$ be given explicitly as sums of monomials, and define $P(x) = 1 + \prod_{i=1}^{k}(P_i(x) + 1)$. We have then $\deg(P) = \sum_{i=1}^{k} \deg(P_i)$, and clearly $P(x) = 0$ if and only if $P_i(x) = 0$, for $i \in [n]$. Suppose that $\deg(P) < n$, and that we are given $a \in \mathbb{F}_2^n$ such that $P(a) = 0$. Then the task is to find $a' \neq a$ such that $P(a') = 0$. We call this problem Explicit Chevalley, and Papadimitriou has shown [22] that it is in PPA.

Could it be that Explicit Chevalley is PPA-complete? We find this highly unlikely. There are two restrictions on the input of Explicit Chevalley. Firstly, the polynomial P is given by an arithmetic circuit (in fact by an arithmetic formula) of specific form. Secondly, and more importantly, the number of variables not only upper bounds the multilinear degree of P, but also the degree of P. The first restriction can be easily relaxed. We can define and compute recursively very easily the circuit degree (also known as the formal degree; see Section 2.3) of the arithmetic circuit which is an upper bound on the degree of the polynomial computed by the circuit. Could it be that the problem, specified by an arithmetic circuit whose circuit degree is less than n, becomes PPA-complete? While this problem might be indeed harder than Explicit Chevalley, we still don't think that it is PPA-complete.

We believe that the more important restriction in Papadimitriou's problem is the one on the degree of the polynomial P computed by the input circuit. As we have seen, to have an even number of zeros, mathematically it is only required that the multilinear degree of P is less than n, so putting the restriction on the degree of P is too stringent. Let's try then to consider instances specified by arithmetic circuits computing polynomials of multilinear degree less than n. However, here we face a serious difficulty. We can't just promise that the polynomial has multilinear degree less than n since PPA is a syntactic class. We must be able to verify syntactically that it is indeed the case.

The multilinear degree of the polynomial is decided by the parity of the monomials computed by the circuit which contain every variable. Let us call such monomials *maximal*. Indeed, the multilinear degree of P is less than n if and only if an even number of maximal

monomials are computed by the circuit. A very general way to prove efficiently that a set is of even cardinality is to give a polynomial Turing machine which computes a perfect matching on the elements of the set. However, the parsing of monomials in arbitrary arithmetic circuits is a rather complex task [19]. For a start, the number of maximal monomials computed by a polynomial size arithmetic circuit can be doubly exponential, making even the description of such a monomial impossible in polynomial time. Fortunately, the situation over the field $\mathbb{F}_2$ simplifies a lot, thanks to cancellations due to certain symmetries. In fact, we are able to show that over $\mathbb{F}_2$ it is sufficient to consider only those monomials which are computed by consistent left/right labellings of the sum gates participating in the computation of the monomial, because the rest of the monomials cancel out. We call such labellings *parse subcircuits*, and we call those parse subcircuits which compute maximal monomials *maximal*. The introduction of parse subcircuits was inspired by the concept of parse trees in [16, 20]. Technically, this results shows that that computing the multilinear degree is in $\oplus$P, the complexity class Parity P.

Is there a chance that for a general circuit computing the multilinear degree is in P? As it turns out not, unless $\oplus\mathrm{P} = \mathrm{P}$, because we can show that computing the multilinear degree is also $\oplus$P-hard. Therefore we have to identify a restricted class of circuits computing polynomials of even multilinear degree which satisfy two properties: the class is on the one hand restricted enough that we are able to construct a polynomial time perfect matching for the maximal parse subcircuits, but it is also large enough that finding another zero for the circuit is PPA-hard. The main contribution of this paper is that we identify such a class of arithmetic circuit which we call PPA-*circuits*.

The definition of these circuits is inspired by a rather straightforward translation of Papadimitriou's basic PPA-problem into a problem for arithmetic circuits. In a nutshell, the basic PPA-problem is the following. Given a degree-one vertex of a graph, in which every vertex has degree at most two, find another degree-one vertex. Here, the graph, whose vertices are the 0-1 strings of given length, is given via a polynomial time Turing machine M determining the neighbourhood of any specified node. We construct an arithmetic circuit over $\mathbb{F}_2$ which, given a vertex v in this graph, computes the opposite parity of the number of v's neighbours. Therefore, finding another degree-one vertex is then just the same as finding another zero of the polynomial computed by the circuit. Most importantly, the circuit is constructed to be in a special form, which allows for a polynomial-time-computable perfect matching over its maximal parse subcircuits. Roughly speaking, from the Turing machine M that describes the neighbours of vertices, we extract two arithmetic circuits D and F that also describe the neighbours in a certain way. We then define the so-called PPA-*composition* of these two circuits, which produces a circuit C that accesses D and F in a black box fashion. Symmetries of the PPA-composition, reflecting the special structure of degree computation, enable us to construct a polynomial-time-computable perfect matching over its maximal parse subcircuits (cf. Lemma 8). Finally we define a PPA-*circuit* as the sum of a PPA-composition and another circuit whose circuit degree is less than n. This is just a minor extension of the family of PPA-compositions since circuits with degree less than n don't have maximal parse subcircuits. The reason for considering this extended family is that this way our result immediately generalizes Papdimitrou's result [22] about Explicit Chevalley, and it makes also easier to express the equivalence between the algorithmic versions of the Chevalley-Warning theorem and the CNSS.

The definition of our problems, PPA-Circuit-CNSS and PPA-Circuit-Chevalley, is therefore the following. In both cases we are given an n-variable, PPA-circuit C over $\mathbb{F}_2$ and an element $a \in \mathbb{F}_2^n$. In the case of PPA-Circuit Chevalley, a is a zero of C, and for

PPA-Circuit CNSS, we consider the sum of the circuits C and L_a, where L_a is a simple *Lagrange-circuit* having a as its only zero and having a single maximal parse subcircuit. The computational task is to compute another zero of C in case of PPA-Circuit Chevalley, and a satisfying assignment for $C + L_a$ in case of PPA-Circuit CNSS. Our result is then stated in the following theorem.

▶ **Theorem 1.** *The problems* PPA-Circuit CNSS *and* PPA-Circuit Chevalley *are* PPA*-complete.*

Since the two problems are easily interreducible, for the proof of Theorem 1 we will show that PPA-Circuit CNSS is PPA-easy and PPA-Circuit Chevalley is PPA-hard. For the easiness part we define a graph, inspired by Papadimitriou's construction, whose vertices are the assignments for the variables and the parse subcircuits. There is an edge between a parse subcircuit and an assignment if the monomial defined by the subcircuit takes the value 1 on the assignment. In addition, we also put an edge between two maximal parse subcircuits of the PPA-composition part of the circuit if they are paired by the perfect matching. As it turns out, the odd degree vertices in this graph are exactly the assignments where the polynomial defined by the circuit is 1, and the unique maximal parse subcircuit of the Lagrange-circuit. Technically, the main part of the proof is to give, for every assignment, a polynomial time computable pairing between its exponentially many neighboring parse subcircuits. For the hardness part (which is much simpler to prove) we express the basic PPA-complete problem as a PPA-composition, as we explained above.

1.3 Previous work

Papadimitriou has proven that Explicit Chevalley is in PPA. Varga [27] has shown the same for the special case of CNSS where the input polynomial P is specified as the sum of a polynomial number of polynomials P_i, where each P_i is the product of explicitly given polynomials whose sum of degrees is at most n. In addition, the input also contains a polynomial time computable matching for all but one of the monomials $x_1 \cdots x_n$ of P. However, the paper doesn't address the question why this doesn't make the problem a promise problem. Concerning the hardness of CNSS, Alon proved in [3] the following result. Let P be specified by an arithmetic circuit in a way that it can be checked efficiently that its multilinear degree is n. If a polynomial time algorithm can find a point a where $P(a) = 1$, then there are no one-way permutations.

1.4 Structure of the paper

In Section 2 we recall the definition of the class PPA, the Combinatorial Nullstellensatz and the Chevalley-Warning Theorem, and arithmetic circuits. In Section 3 we define the parse subcircuits of an arithmetic circuit over $\mathbb{F}_2$, and in Proposition 6 we prove that the polynomial computed by the circuit is the sum of the monomials computed by the parse subcircuits. In Section 4 we define PPA-circuits, and in Lemma 8 we prove that in such circuits a perfect matching for the maximal parse subcircuits can be computed in polynomial time. In Section 5 we state the problems PPA-Circuit CNSS and PPA-Circuit Chevalley over $\mathbb{F}_2$ and observe that they are polynomially interreducible. In Section 6 in Theorem 11 we prove that PPA-Circuit CNSS is in PPA, and in Section 7 in Theorem 13 we prove that PPA-Circuit Chevalley is PPA-hard.

2 Preliminaries

2.1 Total functional NP and the class PPA

We denote the set $\{1, \ldots, n\}$ by $[n]$. A polynomially computable binary relation $R \subseteq \{0,1\}^* \times \{0,1\}^*$ is called balanced if for some polynomial $p(n)$, for every x and y such that $R(x,y)$ holds, we have $|y| \leq p(|x|)$. Such a relation defines an NP-*search problem* Π_R whose input is x, and the task is to find for inputs x, where $R(x,y)$ holds for some y, such a *solution* y, and report "failure" otherwise. The class FNP of *functional* NP consists of NP-search problems. For two problems Π_R and Π_S in FNP, we say that Π_R is *reducible to* Π_S if there exist two functions f and g computable in polynomial time such that for every positive x, $S(f(x), y)$ implies $R(x, g(x,y))$.

An NP-search problem is *total* if for every x, there exists a solution y. The class of these problems is called TFNP (for Total Functional NP) by Megiddo and Papadimitriou [21]. Problems in TFNP exhibit very interesting complexity properties. An FNP-complete search problem can not be total unless NP = coNP. It is also unlikely that every problem in TFNP could be solved in polynomial time since this would imply P = NP ∩ coNP. TFNP is a semantic complexity class, in the sense that it involves a promise about the totality of the relation R. It is widely believed that such a promise can not be enforced syntactically on a Turing machine, in fact there is no known recursive enumeration of Turing machines that compute total search problems. As usual with semantic complexity classes, TFNP doesn't seem to have complete problems. On the other hand, several syntactically defined subclasses of TFNP with a rich structure of complete problems have been identified along the lines of the mathematical proofs establishing the totality of the defining relation.

The parity argument subclasses of TFNP were defined by Papadimitriou [22, 23]. They can be specified via concrete problems, by closure under reduction. The Leaf problem is defined as follows. The input is a triple (z, M, ω) where z is a binary string and M is the description of a polynomial time Turing machine[1] that defines a graph $G_z = (V_z, E_z)$ as follows. The set of vertices is $V_z = \{0,1\}^{p(|z|)}$ for some polynomial p. For any vertex $v \in V_z$, the machine M outputs on (z, v) a set of at most two vertices. Then, we define G_z as a graph without self-loops, where $\{v, v'\} \in E_z$ for $v \neq v'$, if $v' \in M(z, v)$ and $v \in M(z, v')$. Obviously G_z is an undirected graph where the degree of each vertex is at most 2, and therefore the number of leaves, that is of degree one vertices, is even. Finally $\omega \in V_z$ is a degree one vertex that we call the *standard leaf*. The output of the problem Leaf is a leaf of G_z different from the standard leaf. The Polynomial Parity Argument class PPA is the set of total search problems reducible to Leaf. The directed class PPAD is defined by D-Leaf, the directed analog of Leaf. In the problem D-Leaf the Turing machine defines a directed graph, where the indegree and outdegree of every vertex is at most one. The standard leaf ω is a source, and the output is a sink or source different from the ω.

As shown in [23], the definition of PPA can capture also those problems for which the underlying graph has unbounded degrees and we are seeking for another odd-degree vertex. Specifically, suppose there exists a polynomial time *edge recognition* algorithm $\epsilon(v, v')$, which decides whether $\{v, v'\} \in E_z$. Assume also, that in addition we have a polynomial time *pairing function* $\phi(v, w)$, where by definition, for every vertex v, the function $\phi(v, \cdot)$ satisfies the following properties. For every even degree vertex v, it is a pairing between the vertices adjacent to v, that is for every such vertex w, we have $\phi(v, w) = w'$, where $w' \neq w$, w' is also

[1] The requirement for M to run in polynomial can be imposed by adding a clock.

adjacent to v, and $\phi(v, w') = w$. For odd degree vertices v, we have exactly one adjacent vertex w such that w is mapped to itself, and on the remaining adjacent vertices it is pairing as in the case of an even degree vertex v. The input also contains an odd degree vertex v with a proof for that, in the form of an adjacent vertex w, such that $\phi(v, w) = w$. In [23, Corollary to Theorem 1], Papadimitriou showed that any problem defined in terms of an edge recognition algorithm and a pairing function is in PPA.

2.2 Combinatorial Nullstellensatz and Chevalley-Warning Theorem

Let $\mathbb{F}$ be a field. An *polynomial over* $\mathbb{F}$ (or shortly a polynomial) in n variables is a formal expression $P(x) = P(x_1, \ldots, x_n)$ of the form

$$P(x_1, \ldots, x_n) = \sum_{d_1, \ldots, d_n \geq 0} c_{d_1, \ldots, d_n} x_1^{d_1} \cdots x_n^{d_n},$$

where the coefficients $c_{d_1,\ldots,d_n}$ are from $\mathbb{F}$, and only a finite number of them are different from zero. The *degree* $\deg(P)$ of P is the largest value of $d_1 + \cdots + d_n$ for which the coefficient $c_{d_1,\ldots,d_n}$ is non-zero, where by convention the degree of the zero polynomial is $-\infty$. The ring of polynomials over $\mathbb{F}$ in n variables is denoted by $\mathbb{F}[x_1, \ldots, x_n]$.

Every polynomial $P \in \mathbb{F}[x_1, \ldots, x_n]$ defines naturally a function from $\mathbb{F}^n$ to $\mathbb{F}$. While over infinite fields this application is one-to-one, this is not true over finite fields where different polynomials might define the same function. For example, over the field $\mathbb{F}_q$ of size q, the polynomial $x^q - x$ is not the zero polynomial (it has degree q), but it computes the zero function.

Numerous results are known about the properties of zero sets of polynomials. The Combinatorial Nullstellensatz of Alon [2] is a higher dimensional extension of the well known fact that a non-zero polynomial of degree d has at most d zeros. It was widely used to prove a variety of results, among others, in combinatorics, graph theory and additive number theory.

▶ **Theorem 2** (Combinatorial Nullstellensatz). *Let $\mathbb{F}$ be a field, let $d_1, \ldots, d_n$ be non-negative integers, and let $P \in \mathbb{F}[x_1, \ldots, x_n]$ be a polynomial. Suppose that $\deg(P) = \sum_{i=1}^n d_i$, and that the coefficient of $x_1^{d_1} \cdots x_n^{d_n}$ is non-zero. Then for all subsets $S_1, \ldots, S_n$ of $\mathbb{F}$ with $|S_i| > d_i$, for $i = 1, \ldots, n$, there exists $(s_1, \ldots s_n) \in S_1 \times \cdots \times S_n$ such that $P(s_1, \ldots, s_n) \neq 0$.*

The classical result of Chevalley [12] and Warning [28] asserts that if the sum of degrees of some polynomials is less than the number of variables, than the number of their common zeros is divisible by the characteristic of the field.

▶ **Theorem 3** (Chevalley-Warning Theorem). *Let $\mathbb{F}$ be a field of characteristic p, and let $P_1, \ldots, P_k \in \mathbb{F}[x_1, \ldots, x_n]$ be non-zero polynomials. If $\sum_{i=1}^k \deg(P_i) < n$, then the number of common zeros of $P_1, \ldots, P_k$ is divisible by p. In particular, if the polynomials have a common zero, they also have another one.*

Both of these results clearly suggest a computational problem in TFNP: Given a (set of) polynomial(s) satisfying the respective condition of these theorems, find an element in $\mathbb{F}^n$ satisfying the respective conclusion. We study here these problems over the two-element field $\mathbb{F}_2$ where both theorems have a particularly simple form, in fact they become almost the same statement. To see that, let us recall that a *multilinear polynomial* is a polynomial of the form $M(x_1, \ldots, x_n) = \sum_{T \subseteq \{1,\ldots,n\}} c_T x_T$, where x_T stands for the monomial $\prod_{i \in T} x_i$, and the coefficients c_T are elements of $\mathbb{F}_2$. We say that a monomial x_T is *in* M if $c_T = 1$. The degree of a multilinear polynomial M is the cardinality of the largest set T such that x_T is

in M. It is well known that for every polynomial P over $\mathbb{F}_2$, there exists a unique multilinear polynomial $M_P(x_1, \ldots, x_n)$ such that P and M_P compute the same function. We define the *multilinear degree* of a polynomial P over $\mathbb{F}_2$ by $\mathrm{mdeg}(P) = \deg(M_P)$. We call a monomial *maximal* if its multilinear degree is n. Clearly $\mathrm{mdeg}(P) \leq \deg(P)$, and $\mathrm{mdeg}(P) = n$ if and only if the number of maximal monomials of P is odd. Using the notion of multilinear degree, we can now state the rather simple equivalent formulations of the above theorems over $\mathbb{F}_2$.

▶ **Theorem 4** (Combinatorial Nullstellensatz over $\mathbb{F}_2$)**.** *Let $P \in \mathbb{F}_2[x_1, \ldots, x_n]$ be a polynomial such that $\mathrm{mdeg}(P) = n$. Then there exists $a \in \mathbb{F}_2^n$ such that $P(a) = 1$.*

▶ **Theorem 5** (Chevalley-Warning Theorem over $\mathbb{F}_2$)**.** *Let $P \in \mathbb{F}_2[x_1, \ldots, x_n]$ be a polynomial such that $\mathrm{mdeg}(P) < n$, and let $a \in \mathbb{F}_2^n$ such that $P(a) = 0$. Then there exists $b \neq a$ such that $P(b) = 0$.*

2.3 Arithmetic circuits

An n-variable, m-output *arithmetic circuit* C over a field $\mathbb{F}$ is a vertex-labeled, acyclic directed graph whose vertices are called *gates*. It has n *variable gates* of in-degree 0, labeled by the variables $x_1, \ldots, x_n$. There is at most one *constant gate* of in-degree 0, labeled by the constant, for each non-zero field element. The variable and constant gates are called *input gates*. The other gates are of in-degree 2, and are called *computational gates*. They are labeled by $+$ or $\times$, the former are the *sum gates*, and the latter the *product gates*. The number of computational gates of out-degree 0 is m, and they are called the *output gates*.

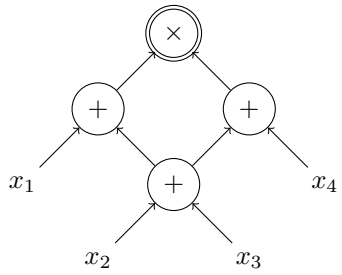


Figure 1 A 4-variable, single-output arithmetic circuit.

For a computational gate g, we distinguish its two children, by specifying the *left* and the *right* child. The left child is denoted by g_ℓ and the right child by g_r. We denote the set of sum gates by G^+, and the set of product gates by $G^\times$. The *size* of C is the number of its gates, and the *depth* of C is the length of the longest path from an input gate to an output gate.

The definition of an arithmetic circuit can be extended naturally to include computational gates of in-degree different from 2. Unary computational gates by definition act as the identity operator. The children of computational gates of in-degree $k > 2$ are distinguished by some some distinct labeling over some set of size k. It is easy to see that such an extended circuit can be simulated by a circuit with binary computational gates, which computes the same polynomial, and has only a polynomial blow-up in size. Our default circuits will be with binary computational gates, and we will mention explicitly when this is not the case.

A *subcircuit* of a circuit C is a subgraph of C which is also a circuit. The *subcircuit rooted at* gate g is the subgraph induced by all vertices contained on some path from the input gates to g, it will be denoted by C_g. The *left subcircuit of* C, denoted by C_ℓ, is the subcircuit

rooted at the left child of the root of C, and the *right subcircuit* C_r is defined similarly. The *composition* of arithmetic circuits is defined in a natural way. If C_1 is an n-variable, m-output circuit and C_2 is a k-variable, n-output circuit then $C_1 \circ C_2$ is the k-variable, m-output circuit composed of C_1 and C_2 where the output gates of C_1 are identified with the variable gates of C_2, and the identical constant gates of the two circuits are also identified. Let C_1 and C_2 be n-variable, single-output arithmetic circuit. The *disjoint sum* $C_1 \oplus C_2$ of C_1 and C_2 is the n-variable, single-output arithmetic circuit whose output gate is a sum gate, its left and right subcircuits are disjoint copies of C_1 and C_2 except for the input gates that C_1 and C_2 share. The disjoint sum naturally generalizes to more than two circuits.

Every gate g in an arithmetic circuit computes an n-variable polynomial $P_g(x)$ in the natural way, which can be defined by recursion on the depth of the gate. An input gate g labeled by $\alpha \in \{x_1, \ldots, x_n\} \cup \mathbb{F}$ computes $P_g = \alpha$. If $g \in G^+$ then $P_g = P_{g_\ell} + P_{g_r}$, if $g \in G^\times$ then $P_g = P_{g_\ell} P_{g_r}$. The polynomial computed by a single-output arithmetic circuit C is the polynomial computed by its output gate, which we will denote by $C(x)$. We define similarly by recursion the *circuit degree* $\mathrm{cdeg}(C)$ of C. If an input gate g is labeled by $\alpha \in \mathbb{F}$ then $\mathrm{cdeg}(C_g) = 0$, and if it is labeled by $\alpha \in \{x_1, \ldots, x_n\}$ then $\mathrm{cdeg}(C_g) = 1$. For computational gates, if $g \in G^+$ then $\mathrm{cdeg}(C_g) = \max\{\mathrm{cdeg}(C_{g_\ell}), \mathrm{cdeg}(C_{g_r})\}$, and if $g \in G^\times$ then $\mathrm{cdeg}(C_g) = \mathrm{cdeg}(C_{g_\ell}) + \mathrm{cdeg}(C_{g_r})$. The circuit degree can be computed in polynomial time, and we clearly have $\deg(C(x)) \leq \mathrm{cdeg}(C)$.

Over the base field $\mathbb{F}_2$, we call an element $a \in \mathbb{F}_2^n$, such that $C(a) = 1$, a *satisfying assignment* for C, and an element a, such that $C(a) = 0$, a *zero* of C. For every $a \in \mathbb{F}_2^n$, we define the *Lagrange-circuit* L_a as $C_1 \times \cdots \times C_n$, where $C_i = x_i$ if $a_i = 1$, and $C_i = x_i + 1$ if $a_i = 0$. Clearly $\mathrm{mdeg}(L_a(x)) = n$, and the only satisfying assignment for L_a is a.

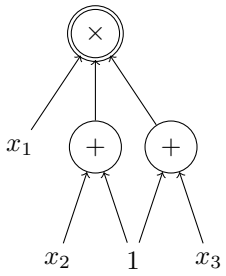


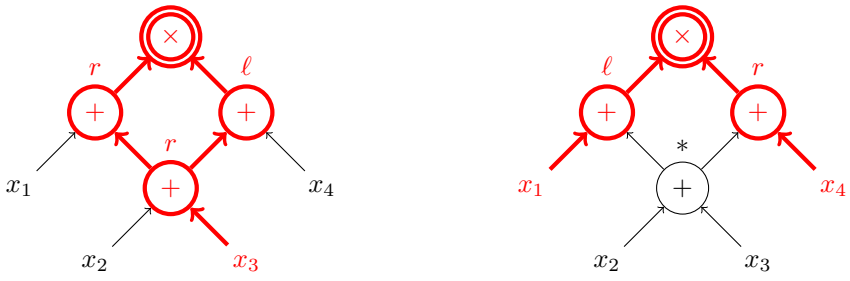
Figure 2 Lagrange-circuit L_{100}.

3 Parse subcircuits

We would like to understand how monomials are computed by a single-output arithmetic circuit C. If g is a sum gate, then the set of monomials computed by C_g is a subset of the union of the set of monomials computed by C_{g_ℓ} and by C_{g_r}. If g is a multiplication gate, then every monomial computed by C_g is the product of a monomial computed by C_{g_ℓ} and a monomial computed by C_{g_r}. A marking of the gates in G^+ from the set $\{\ell, r\}$ therefore computes naturally a monomial of $C(x)$. At first sight it seems that by considering markings restricted to the sum gates effectively participating in the computing of the monomial, we could compute all of them. This is in fact the case when the fanout of every sum gate is one, but this is not true in general circuits since the sum gates can be used several times in the computation of a monomial with possibly inconsistent markings. However, as we show it below, this is essentially true over fields of characteristic 2, where it is sufficient to consider only consistent markings. By doing that, we have to be careful about two things: when

computing a monomial by some marking, we shouldn't mark those sum gates which don't participate in its computation. Indeed, by considering the two possible markings also for irrelevant gates, we would assure that the monomial is necessarily computed an even number of times, making the whole process false. On the other hand, we should mark all the sum gates necessary for the computation of the monomial. We make all this precise by the notion of closed marking and parse subcircuit.

Let C be a single-output arithmetic circuit. A *marking* of C is a partial function $S\colon G^+ \to \{\ell, r\}$, from the sum gates of C to the marks $\{\ell, r\}$. We can equivalently specify a marking by a total function $S^*\colon G^+ \to \{\ell, r, *\}$ where $S^*(g) = *$ if and only if $S(g)$ is undefined. We denote by $\mathrm{Dom}(S)$ the domain of S. For the output gate of C, let S_ℓ be the restriction of S to the sum gates in C_ℓ and let S_r be the restriction of S to the sum gates in C_r. We define $G_S = (V_S, E_S)$, the *accessibility graph* of S by induction on the depth of C. If C is a single vertex then V_S consists of this vertex, and $E_S = \emptyset$. Otherwise, if the output gate is a product gate, then V_S consists of the output gate of C added to $V_{S_\ell} \cup V_{S_r}$, and E_S consists of the two edges from the two children of the output gate to the output gate, added to $E_{S_\ell} \cup E_{S_r}$. If the output gate of C is a sum gate with mark ℓ then V_S consist of the output gate of C added to V_{S_ℓ}, and E_S consists of the edge from the left child of the output gate to the output gate, added to E_{S_ℓ}. The definition in the case when the mark of the output gate is r is analogous. If the output gate of C doesn't have a mark then the accessibility graph is just this single node.



(a) parse subcircuit computing x_3^2 **(b)** parse subcircuit computing x_1x_4

Figure 3 Two parse subcircuits for Figure 1, note that the second one doesn't access all sum gates.

We say that a marking S is *closed* if $\mathrm{Dom}(S) = V_S \cap G^+$, that is if the accessible sum gates of C are exactly those where S is defined. If S is closed then the accessibility graph G_S, with the vertex labels inherited from C, is in fact a subcircuit of C. The inclusion $\mathrm{Dom}(S) \subseteq V_S \cap G^+$ ensures that the only node of out-degree 0 in G_S is the output gate of C, and the inclusion $V_S \cap G^+ \subseteq \mathrm{Dom}(S)$ ensures that the leaves of G_S are leaves in C. We call this subcircuit the *parse subcircuit* induced by S, and denote it by C_S. The set of parse subcircuits of C will be denoted by $\mathcal{S}(C)$. Observe that a parse subcircuit has binary product gates but unary sum gates which act as the identity operator. The polynomial $C_S(x)$ *computed by* the parse subcircuit C_S is therefore a monomial, which we denote by $m_S(x)$. We say that a parse subcircuit C_S is *maximal* if the multilinear degree of $m_S(x)$ is n, that is $m_S(x) = x_1 \cdots x_n$. We say that two parse subcircuits C_S and $C_{S'}$ are *consistent* if for every $g \in \mathrm{Dom}(S) \cap \mathrm{Dom}(S')$, we have $S(g) = S'(g)$.

Clearly, the mapping from closed markings to induced parse subcircuits is a bijection. Therefore, to ease notation, we will often call the closed marking S itself the parse subcircuit, and we will speak about the gates, subcircuits and other circuit related notions of S, instead

of C_S. The notation used for the monomial computed by a parse subcircuit is already consistent with this convention.

▶ **Proposition 6.** *Let C be a single-output arithmetic circuit over a field $\mathbb{F}$ of characteristic 2. Then*

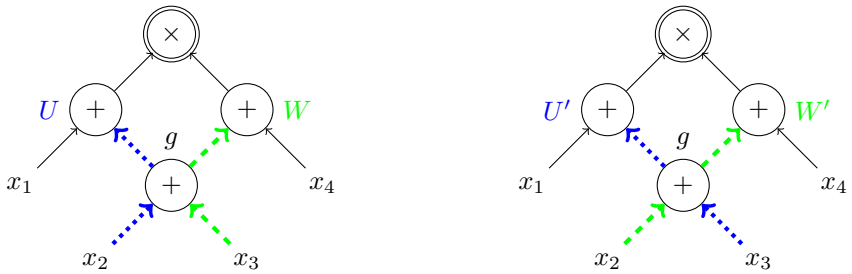
$$C(x) = \sum_{S \in \mathcal{S}(C)} m_S(x).$$

Proof. We prove by induction on the depth of the circuit. If C consists of a single gate, the statement is obvious.

Otherwise, the parse subcircuits of $\mathcal{S}(C_\ell)$ (respectively $\mathcal{S}(C_r)$) are exactly the parse subcircuits of $\mathcal{S}(C)$ restricted to the sum gates of C_ℓ (respectively C_r). When the output gate of C is a sum gate then conversely, $\mathcal{S}(C)$ can be obtained from $\mathcal{S}(C_\ell) \cup \mathcal{S}(C_r)$ by extending the markings in the latter set with the appropriate mark for the root of C. Therefore, using the definitions of $C(x)$ and $m_S(x)$, we get

$$\begin{aligned} C(x) &= C_\ell(x) + C_r(x) \\ &= \sum_{S \in \mathcal{S}(C_\ell)} m_S(x) + \sum_{S \in \mathcal{S}(C_r)} m_S(x) \\ &= \sum_{S \in \mathcal{S}(C),\ S(\text{root})=\ell} m_{S_\ell}(x) + \sum_{S \in \mathcal{S}(C),\ S(\text{root})=r} m_{S_r}(x) \\ &= \sum_{S \in \mathcal{S}(C)} m_S(x), \end{aligned}$$

where the second equality comes from the inductive hypothesis.



(a) inconsistent U, W **(b)** inconsistent U', W'

Figure 4 The involutive pair $(U, W) \leftrightarrow (U', W')$ in the proof of Proposition 6 with $m_U m_W = x_2 x_3 = m_{U'} m_{W'}$ contributes zero to $C(x)$.

When the output gate of C is a product gate, the situation is more complicated. The parse subcircuits S_ℓ and S_r are always consistent for $S \in \mathcal{S}(C)$, but an arbitrary parse subcircuit $U \in \mathcal{S}(C_\ell)$ is not necessarily consistent with an arbitrary parse subcircuit $W \in \mathcal{S}(C_r)$. Therefore the crux of the induction step is to show that the contribution of $m_U(x)m_W(x)$ to $C(x)$ is zero when we sum over all inconsistent U and W. Indeed, we claim that

$$\sum_{(U,W) \in \mathcal{S}(C_\ell) \times \mathcal{S}(C_r),\ U,W \text{inconsistent}} m_U(x) m_W(x) = 0.$$

To prove this, we define an involution $(U, W) \leftrightarrow (U', W')$ over inconsistent pairs in $\mathcal{S}(C_\ell) \times \mathcal{S}(C_r)$ such that $m_U(x)m_W(x) + m_{U'}(x)m_{W'}(x) = 0$. For this let us fix some

topological ordering of the gates in C with respect to the edges of the circuit, and let g be the first sum gate in this ordering where U and W have different marks, say $U(g) = \ell$ and $W(g) = r$. Let the restriction of U to the sum gates of C_g be T_0 and let the restriction of W to the sum gates of C_g be T_1. Both T_0 and T_1 are parse subcircuits in C_g, which are inconsistent only at g. Also, for some monomials $m_0(x)$ and $m_1(x)$, we have $m_U(x) = m_0(x)m_{T_0}(x)$ and $m_W(x) = m_1(x)m_{T_1}(x)$. The parse subcircuit U' is obtained from U by exchanging inside C_g the parse subcircuit T_0 for the parse subcircuit T_1, that is $U' = (U \setminus T_0) \cup T_1$. The parse subcircuit W' is similarly defined from W with the roles of T_0 and T_1 reversed. It follows from the choice of g that U' and W' are parse subcircuits respectively in $\mathcal{S}(C_\ell)$ and $\mathcal{S}(C_r)$ such that the first inconsistency between them in the topological order is at g. Therefore starting the same process with (U', W') we obtain (U, W), and thus the mapping is indeed an involution. Since $m_{U'}(x) = m_0(x)m_{T_1}(x)$ and $m_{W'}(x) = m_1(x)m_{T_0}(x)$, we can conclude that $m_U(x)m_W(x) + m_{U'}(x)m_{W'}(x) = 0$.

We can now complete the induction step for product gates by observing the equalities

$$\begin{aligned}
C(x) &= C_\ell(x) \times C_r(x) \\
&= \left(\sum_{U \in \mathcal{S}(C_\ell)} m_U(x) \right) \times \left(\sum_{W \in \mathcal{S}(C_r)} m_W(x) \right) \\
&= \sum_{(U,W) \in \mathcal{S}(C_\ell) \times \mathcal{S}(C_r),\ U,W \text{consistent}} m_U(x) m_W(x) \\
&= \sum_{S \in \mathcal{S}(C)} m_{S_\ell}(x) m_{S_r}(x) \\
&= \sum_{S \in \mathcal{S}(C)} m_S(x).
\end{aligned}$$

◀

Though it is not directly related to the main result of the paper, we prove here, essentially as a corollary of the previous proposition, that deciding if the polynomial computed by a circuit over the two elements field has maximal multilinear degree is $\oplus$P-complete. Note that by the Chevalley-Warning theorem, the multilinear degree of a circuit is maximal if and only if it has odd number of satisfying assignments, and via this correspondence Proposition 7 can also be proved by using the number of 1's to build a balanced relation. The point of our proof of Proposition 7 is to show this without referring to the Chevalley-Warning theorem, and therefore illustrate the use of maximal parse subcircuits.

▶ **Proposition 7.** *Let C be an n-variable, single-output arithmetic circuit over the field $\mathbb{F}_2$. The problem of deciding if* $\mathrm{mdeg}(C(x)) = n$ *is* $\oplus$P*-complete.*

Proof. For the easiness part, we can define a balanced relation $R(C, S)$ where $S \in \mathcal{S}(C)$, which equals 1 if and only if S is a maximal parse subcircuit. By Proposition 6, we know that the polynomial computed by the circuit C is the sum of all the monomials computed by the parse subcircuits. Among all the parse subcircuits, only the monomials computed by maximal parse subcircuits have degree n. Thus $\mathrm{mdeg}(C(x)) = n$ if and only if there is an odd number of maximal parse subcircuits.

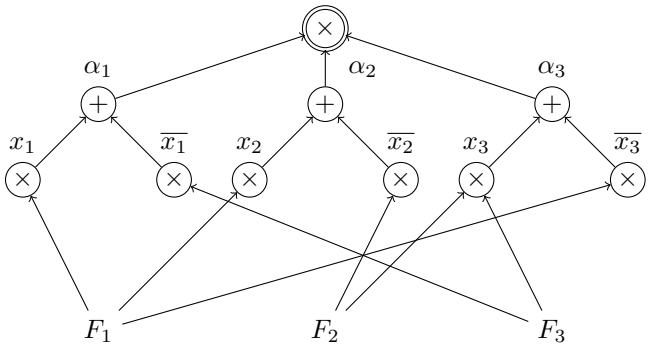


Figure 5 Image of $(x_1 \vee x_2 \vee \overline{x_3}) \wedge (\overline{x_2} \vee x_3) \wedge (x_3 \vee \overline{x_1})$ by the reduction.

For the hardness part, we will reduce the well known $\oplus$P-complete problem $\oplus$3-SAT [26] to the maximality of $\mathrm{mdeg}(C(x))$. Let $\phi = \{F_1, F_2, \ldots, F_m\}$ be an instance of 3-SAT, where the clause F_i is the conjunction of three literals belonging to $\{x_1, \overline{x_1}, \ldots, x_n, \overline{x_n}\}$. The reduction maps ϕ to an m-variable, single-output and depth-3 arithmetic circuit C defined as follows. The output gate at level 0 is a product gate. It has n children $\alpha_1, \ldots, \alpha_n$, all plus gates, which compose the first level of the circuit. At level 2, for all $1 \leq j \leq n$, the gate α_j has two children x_j and $\overline{x_j}$, which are product gates. The gate x_j is the left child of α_j, and $\overline{x_j}$ is its right child. Finally at level 3 are the m variable gates $F_1, \ldots, F_m$, such that F_i is a child of $y \in \{x_1, \overline{x_1}, \ldots, x_n, \overline{x_n}\}$ if $y \in F_i$ in ϕ. The following is an illustration of the circuit which is the image of the formula $(x_1 \vee x_2 \vee \overline{x_3}) \wedge (\overline{x_2} \vee x_3) \wedge (x_3 \vee \overline{x_1})$ by the reduction.

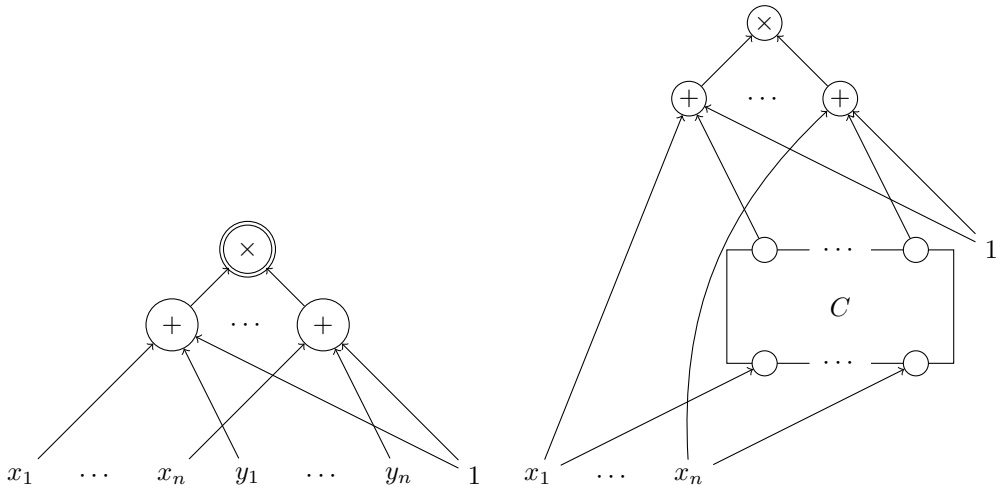
We give a one-to-one mapping S from the assignments of ϕ to the parse subcircuits of $\mathcal{S}(C)$. Since all plus gates of C are reachable from the output gate, a parse subcircuit of C is an $\{\ell, r\}$-marking of the gates $\alpha_1, \ldots, \alpha_n$. The parse subcircuits are therefore naturally identified with the elements of $\{\ell, r\}^n$. For an assignment $x \in \{0,1\}^n$, the map S is defined as

$$S(x)_i = \begin{cases} \ell & \text{if } x_i = 1 \\ r & \text{if } x_i = 0. \end{cases}$$

To finish the proof we show that x is a satisfying assignment if and only if $S(x)$ is a maximal parse subcircuit. To see that, observe that x is a satisfying assignment if and only if each F_i in ϕ contains a true literal. By the definition of S, the clause F_i contains a true literal exactly when the variable F_i of C is in the parse subcircuit $C_{S(x)}$. Since $C_{S(x)}$ is maximal if and only if F_i is in the parse subcircuit $C_{S(x)}$ for all i, this concludes the proof. ◀

4 PPA-circuits

Given an arbitrary circuit C and a satisfying assignment, asking for another satisfying assignment would be an NP-hard problem. We want to restrict the form of the circuit C in a way which takes into consideration the structure of problems in PPA.



(a) The arithmetic circuit I. **(b)** The compound arithmetic circuit $I \diamond C$.

Figure 6 The arithmetic circuits I and $I \diamond C$.

For this, we use repeatedly a $2n$-variable, single-output arithmetic circuit I. The circuit I is of depth 2, its output gate is a product gate with n children, all sum gates. Every sum gate has 3 children, the left child of the ith gate is the variable gate x_i, its center child is the variable gate y_i, and its right child is the constant gate 1. For an n-variable, n-output circuit C, we define $I \diamond C$, the *diamond composition* of I with C, as the n-variable, single-output circuit composed from a circuit I at the top and C below. More precisely, the variable gates of $I \diamond C$ labeled by $x_1, \ldots, x_n$ are also the first n variables of I, and the variable gates $y_1, \ldots, y_n$ of I are identified with the output gates of C. If C has also a constant gate 1, it is identified with the constant gate 1 of I.

The polynomial computed by the circuit I is $I(x_1, \ldots, x_n, y_1, \ldots, y_n) = \prod_{i=1}^{n}(x_i + y_i + 1)$. It is easy to check that $I(x, y)$ is 1 if and only if the two n-bit strings $x_1, \ldots, x_n$ and $y_1, \ldots, y_n$ are equal. Therefore $I \diamond C(x) = 1$ if and only if $C(x) = x$.

Given two n-variable, n-output arithmetic circuits D and F, we consider the set of six n-variable, single-output circuits

$$\mathcal{C}_{D,F} = \{I_1 \diamond D_1 \circ F_1, I_2 \diamond F_2 \circ D_2, I_3 \diamond D_3 \circ D_4, I_4 \diamond D_5, I_5 \diamond F_3 \circ F_4, I_6 \diamond F_5\},$$

where $I_1, \ldots, I_6$ are copies of I; $D_1, \ldots, D_5$ are copies of D; $F_1, \ldots, F_5$ are copies of F, and the six circuits share the same input gates. The PPA-*composition* of D and F is the n-variable, single-output circuit $C_{D,F}$ is the disjoint sum of the six circuits in $\mathcal{C}_{D,F}$. We call the circuits in $\mathcal{C}_{D,F}$ the *components* of $C_{D,F}$. The polynomial computed by $C_{D,F}$ is

$$\begin{aligned} C_{D,F}(x) = {} & I(x, D(F(x))) + I(x, F(D(x))) + I(x, D(D(x))) \\ & + I(x, D(x))) + I(x, F(F(x))) + I(x, F(x))). \end{aligned}$$

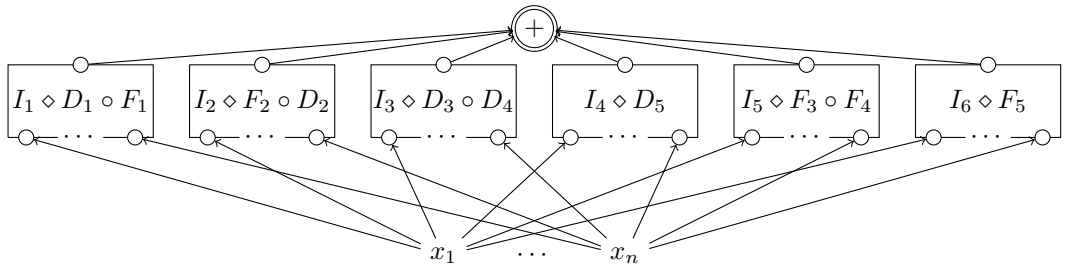


Figure 7 The circuit $C_{D,F}$, the PPA-composition of the circuits D and F.

The main structural property of a PPA-composition C is that it computes a polynomial whose multilinear degree is less than n. Moreover, a witness for that can be computed in polynomial time. By Proposition 6, the multilinear degree of $C(x)$ is determined by the parity of its maximal parse subcircuits, $\mathrm{mdeg}(C(x)) = n$ if and only if the parity of the maximal parse subcircuits is odd. Thus, the multilinear degree of $C(x)$ can be certified by a special type of syntactically defined matching over its maximal parse subcircuits. Formally, a *matching for maximal parse subcircuits in* C is a polynomial time Turing machine μ which defines a matching over the maximal parse subcircuits of C as follows: S and S' are *matched* if $\mu(C, S) = S'$ and $\mu(C, S') = S$. If μ defines a perfect matching between the maximal parse subcircuits, then $\mathrm{mdeg}(C(x)) < n$. If μ defines a perfect matching *outside* some maximal parse subcircuit T, meaning that T is the only maximal parse subcircuit without a matching pair in μ, then $\mathrm{mdeg}(C(x)) = n$.

All the above statements hold also for circuits which are the direct sum of a PPA-composition and another circuit which certifiably has no maximal parse subcircuit. This is obviously the case of circuits which compute polynomials of degree less than n. Our final set of authorized circuits are of this form. We say that a circuit C is a PPA-*circuit* if for some D and F, we have $C = C_{D,F} \oplus C'$, where $\mathrm{mdeg}(C') < n$.

▶ **Lemma 8.** *If C is a* PPA*-circuit then* $\mathrm{mdeg}(C(x)) < n$*, and a perfect matching μ between the maximal parse subcircuits of C can be computed in polynomial time.*

Proof. Let $C = C_{D,F} \oplus C'$ where $\mathrm{mdeg}(C') < n$. We can suppose without less of generality that C' is the empty circuit, that is $C = C_{D,F}$. Since the six components of C are pairwise disjoint (except for the input gates), every maximal parse subcircuit in C consists of the mark of the root of C from the set $\{1, \ldots, 6\}$, and a maximal parse subcircuit in the corresponding component. For the definition of μ we decompose C into the disjoint sum of three circuits C_1, C_2 and C_3 where each of them is the disjoint sum of two PPA-components, and will define the matching inside each of these circuits. The three circuits are as follows: $C_1 = I_1 \diamond D_1 \circ F_1 \oplus I_2 \diamond F_2 \circ D_2$, $C_2 = I_3 \diamond D_3 \circ D_4 \oplus I_4 \diamond D_5$, and $C_3 = I_5 \diamond F_3 \circ F_4 \oplus I_6 \diamond F_5$. Clearly C_2 and C_3 are similar, therefore it is sufficient to define μ for C_1 and C_2.

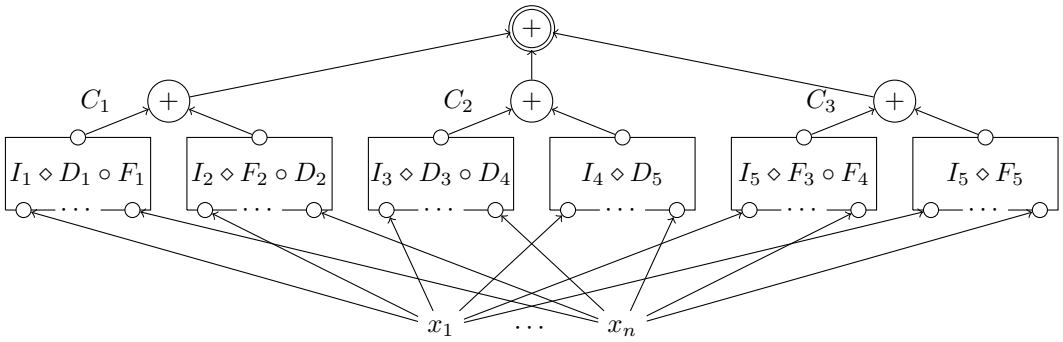


Figure 8 The decomposition $C = C_1 \oplus C_2 \oplus C_3$.

To ease the notation, we rename the subcircuits of C_1 as $I \diamond D \circ F$ and $I' \diamond F' \circ D'$, and we suppose that $I \diamond D \circ F$ is the left subcircuit of C_1 and $I' \diamond F' \circ D'$ is its right subcircuit. Let us denote the output (sum) gate of C_1 by h, the sum gates of I by $h_1, \ldots, h_n$, the output gates of D by $d_1, \ldots d_n$, and the output gates of F by $f_1, \ldots, f_n$. For every gate g in I, D and F, we denote the corresponding gate in I', D' and F' by g', and we also set $h' = h$. Let us recall the h_i has three children, the left child is the input gate x_i, the center child is d_i, the ith output gate of D, and its right child is the constant gate 1. A parse subcircuit can map h_i into one of the three marks ℓ, c and r, corresponding respectively to its left, center, and right child.

We define $\mu(S)$ for the maximal parse subcircuits of $I \diamond D \circ F$, that is when $S(h) = \ell$. The definition for the case $S(h) = r$ is symmetric. Let us first define three sets of indices $S_{\text{out}}, S_{\text{middle}}, S_{\text{in}} \subseteq [n]$. Let $S_{\text{out}} = \{i \in [n] : S(h_i) = c\}$, that is S_{out} contains those indices i for which the edge from the d_i to h_i belongs to S. By definition $i \in S_{\text{middle}}$ if there exists an edge in S from f_i to a gate in D. Finally, $i \in S_{\text{in}}$ if there exists an edge in S from x_i to a gate in F. We claim that $S_{\text{out}} \subseteq S_{\text{in}}$. This is indeed true, since if there exists $i \in S_{\text{out}} \setminus S_{\text{in}}$ then the monomial $m_S(x)$ wouldn't contain the variable x_i, contradicting its maximality. We are now ready to define $S' = \mu(S)$ by distinguishing two cases, depending on if S_{out} is a proper subset of S_{in} or not.

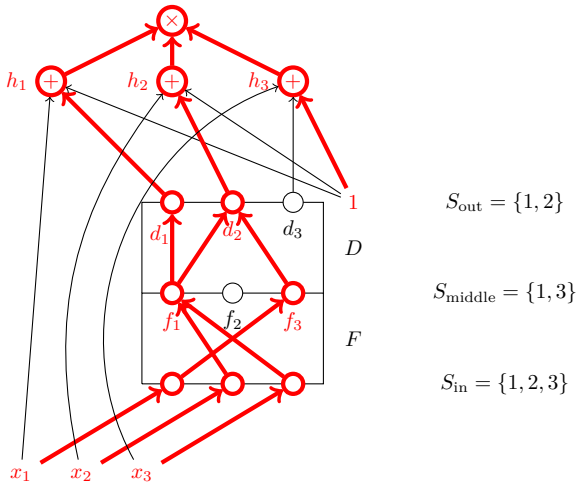
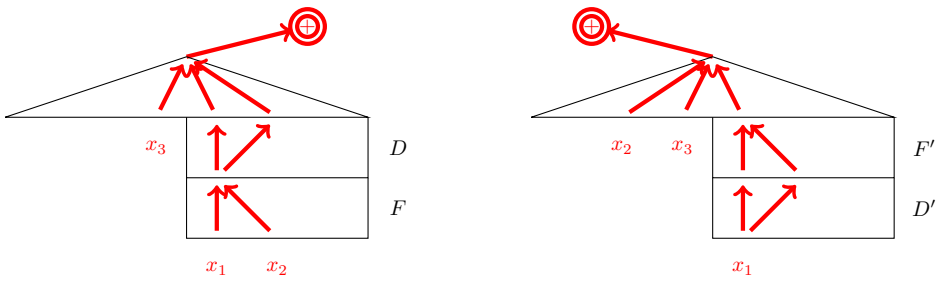


Figure 9 The left subcircuit $I \diamond D \circ F$ of C_1 and the index sets $S_{\text{in}}, S_{\text{middle}}$ and S_{out}.

Case 1: $S_{\text{out}} \subset S_{\text{in}}$. Let i be the smallest index in $S_{\text{in}} \setminus S_{\text{out}}$. By definition, we let S' be the same as S, except on h_i, where S' takes the mark r when $S(h_i) = \ell$, and it takes the mark ℓ when $S(h_i) = r$. This means that the only difference between S and S' is that at the ith sum gate of I, one subcircuit contains the edge from x_i to h_i, whereas the other contains the edge from 1 to h_i. S' is therefore a parse subcircuit. To show that S' is also maximal, the interesting case is when $S(h_i) = \ell$ and $S'(h_i) = r$, that is $m_{S'}(x)$ doesn't directly pick up x_i at h_i. But since $i \in S_{\text{in}}$, the variable x_i is still in S', which is therefore maximal. Finally clearly $\mu(S') = S$.

(a) maximal parse subcircuit S **(b)** maximal parse subcircuit S'

Figure 10 Case 1 of the matching μ for C_1 where i is the smallest index in $S_{\text{in}} \setminus S_{\text{out}}$.



(a) maximal parse subcircuit S **(b)** maximal parse subcircuit S'

Figure 11 Case 2 of the matching μ for C_1: $S_{\text{out}} = S_{\text{in}}$.

Case 2: $S_{\text{out}} = S_{\text{in}}$. In that case first observe that for every index $i \notin S_{\text{out}}$, we have $S(h_i) = \ell$, that is S contains the edge (x_i, h_i), since otherwise $m_S(x)$ wouldn't contain x_i. By definition, let $\text{Dom}(S') = \{g' \in G^+ : g \in \text{Dom}(S)\}$. For the output gate $h' = h$ of C_1 we set $S'(h') = r$, that is S' will be a parse subcircuit of $I' \diamond D' \circ F'$. For the sum gates $h'_1, \ldots, h'_n$ of I, we set $S'(h'_i) = c$ if $i \in S_{\text{middle}}$, and we set $S'(h'_i) = \ell$ otherwise. Finally, for every sum gate $g \in \text{Dom}(S)$ in D or in F, we set $S'(g') = S(g)$.

Let us recall that V_S is the set of vertices of the accessibility graph G_S of S. The proof that S' is a maximal parse subcircuit immediately follows from the following proposition.

▶ **Proposition 9.** *For every computational gate g in $I \diamond D \circ F$, we have*

$$g \in V_S \quad \textit{if and only if} \quad g' \in V_{S'}.$$

Proof. We show the implication from left to right. This is certainly true for the computational gates of I since they are all accessible in G_S, as well as the computational gates of I' in $G_{S'}$.

If $g \in V_S$ is a computational gate of D then there is a path p in G_S from g to h which can be decomposed into $p = p_1 p_2$, where p_1 goes from g to d_i for some $i \in S_{\text{out}}$, and p_2 is the path from d_i to h. In $G_{S'}$ we have therefore a path p'_1 from g' to d'_i. Since $S_{\text{out}} = S_{\text{in}}$, in G_S

we have a path p_3 from x_i to f_j for some $j \in S_{\text{middle}}$. Therefore in $G_{S'}$ there exists a path p'_2 from d'_i to f'_j. Finally, in $G_{S'}$ there is also a path p'_3 from f'_j to h' because $j \in S_{\text{middle}}$. Then $p' = p'_1 p'_2 p'_3$ is a path from g' to h'.

If $g \in V_S$ is a computational gate of F then there is a path p in G_S from g to h which can be decomposed into $p = p_1 p_2 p_3$, where p_1 goes from g to d_i for some $i \in S_{\text{middle}}$, p_2 goes from d_i to f_j for some $j \in S_{\text{out}}$, and p_3 is the path from f_j to h. Then in $G_{S'}$ there exists a path p'_1 from g' to d'_i, and a path p'_2 which goes from d'_i to h' since $i \in S_{\text{middle}}$. Then the path $p' = p'_1 p'_2$ goes from g' to h'.

The implication from right to left follows from the symmetry between S and S'. For this, it is useful to observe that $S'_{\text{out}} = S'_{\text{in}} = S_{\text{middle}}$, and $S'_{\text{middle}} = S_{\text{out}} = S_{\text{in}}$. ◀

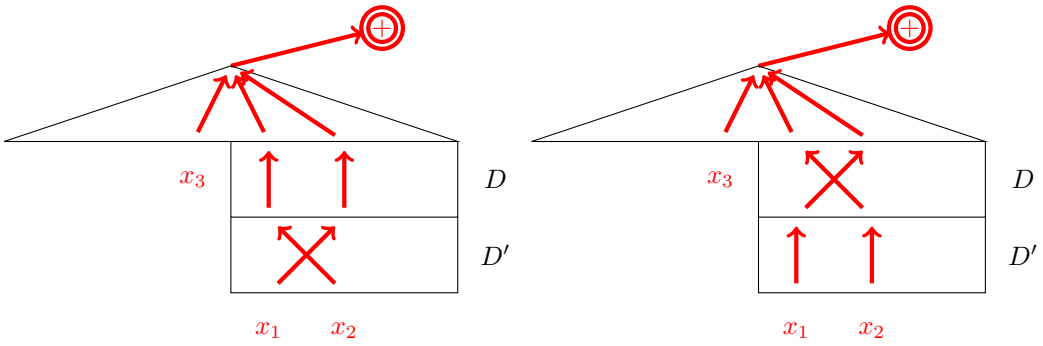
We have $\text{Dom}(S) = V_S \cap G^+$ since S is a parse subcircuit. Proposition 9 and the definition $\text{Dom}(S') = \{g' \in G^+ : g \in \text{Dom}(S)\}$ imply that $\text{Dom}(S') = V_{S'} \cap G^+$, and therefore S' is a parse subcircuit. To prove the maximality of S' let us show that every input gate is in $V_{S'}$. If $i \in S_{\text{middle}}$ then the path p defined above for the computational gates in D yields a path p' from x_i to h'. If $i \notin S_{\text{middle}}$ then the direct path p' from x_i to h' via h'_i exists in $G_{S'}$. Finally μ is clearly involutive in that case too.

We now turn to the description of μ for C_2, where we rename its two subcircuits as $I \diamond D \circ D'$ and $I^* \diamond D^*$. The matching for C_2 has strong analogies with the matching for C_1, to better see this we also use the names I', F and F' respectively for the circuits I, D' and D. This means that $I \diamond D \circ F$ and $I' \diamond F' \circ D'$ are just different names for the circuit $I \diamond D \circ D'$. We suppose that $I \diamond D \circ D'$ is the left subcircuit of C_2 and $I^* \diamond D^*$ is its right subcircuit. Similarly to the circuit C_1, we denote the output gate of C_2 by h, the sum gates of I by $h_1, \ldots, h_n$, the ouput gates of D by $d_1, \ldots d_n$, and the output gates of D' by $d'_1, \ldots, d'_n$. For every gate g in I, D and D', we denote the corresponding gate respectively in I', D' and D by g'. For every gate g in I and D, we denote the corresponding gate in I^* and D^* by g^*. We also set $h^* = h' = h$. Again, h_i has three children, the left child is the input gate x_i, the center child is d_i, the right child is the constant gate 1, and the respective marks are ℓ, c and r.

We first describe $S' = \mu(S)$ when S is a maximal parse subcircuit of $I \diamond D \circ D'$. We define $S_{\text{out}}, S_{\text{middle}}, S_{\text{in}}$ the same way as for the circuit $I \diamond D \circ F$, keeping in mind that $F = D'$. As before, we have $S_{\text{out}} \subseteq S_{\text{in}}$. For the definition of μ we now distinguish three cases.

Case 1: $S_{\text{out}} \subset S_{\text{in}}$. The definition of S' is identical to the first case of the definition of the matching for C_1.

Case 2: $S_{\text{out}} = S_{\text{in}}$ and there exists a sum gate g in D such that $S(g) \neq S(g')$. The definition of S' is identical to the second case of the definition of the matching for C_1, with one exception. The difference is that S' remains in the left subcircuit of C_2, that is for the output gate $h' = h$ we set $S'(h') = \ell$.

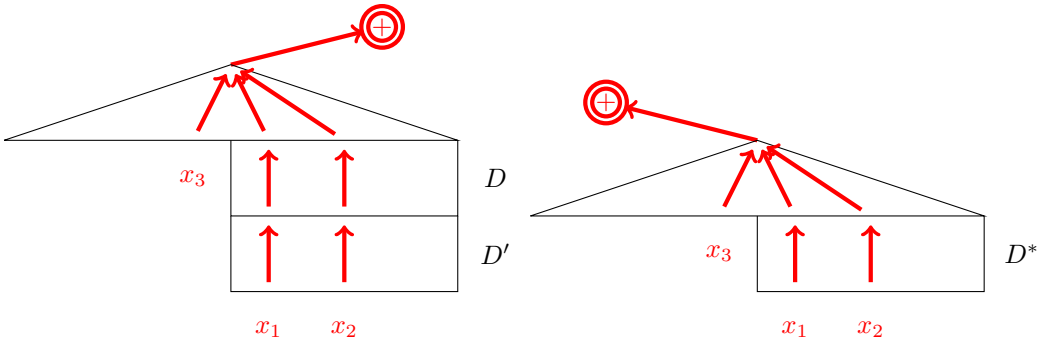


(a) maximal parse subcircuit S **(b)** maximal parse circuit S'

Figure 12 Case 2 of the matching μ for C_2: $S_{\text{out}} = S_{\text{in}}$ and $\exists g, S(g) \neq S(g')$.

Case 3: $S_{\text{out}} = S_{\text{in}}$ and for all sum gate g in D, we have $S(g) = S(g')$. By definition we set $\text{Dom}(S') = \{g^* \in G^+ : g \in \text{Dom}(S)\}$. For the output gate $h^* = h$ of C_2 we set $S'(h^*) = r$, that is S' will be a parse subcircuit of $I^* \diamond D^*$. For every other sum gate $g \in \text{Dom}(S)$, we set $S'(g^*) = S(g)$.

The description $S' = \mu(S)$ when S is a maximal parse subcircuit of $I^* \diamond D^*$ is as follows. By definition we set $\text{Dom}(S') = \{g, g' \in G^+ : g^* \in \text{Dom}(S)\}$. We set $S'(h) = \ell$, that is S' is a parse subcircuit of $I \diamond D \circ D'$. For the sum gates of I, we set $S'(h_i) = S(h_i^*)$. For every sum gate $g^* \in \text{Dom}(S)$ which is in D^*, we set $S'(g) = S'(g') = S(g^*)$.



(a) maximal parse subcircuit S **(b)** maximal parse subcircuit S'

Figure 13 Case 3 of the matching μ for C_2: $S_{\text{out}} = S_{\text{in}}$ and $\forall g, S(g) = S(g')$.

The proof that S' is a maximal parse subcircuit is basically the same as for the case of circuit C_1. It follows immediately from the definition that μ is an involution. The only additional point to see is that in the second case $S' \neq S$ because $S(g) \neq S(g')$, for some gate g in D. ◀

5 The computational problems

We are now ready to define PPA-Circuit CNSS and PPA-Circuit Chevalley, the two computational problems corresponding to the CNSS and to the Chevalley-Warning theorem over $\mathbb{F}_2$. The input will be in both cases an n-variable, single-output PPA-circuit C, and an element $a \in \mathbb{F}_2^n$. In the case of PPA-Circuit Chevalley, it is a zero of C, and

Lemma 8 ensures that C satisfies the hypotheses of the Chevalley-Warning Theorem. For PPA-Circuit CNSS, we consider the circuit $C \oplus L_a$, and Lemma 8 again ensures that this circuit satisfies the hypothesis of the CNSS. The computational task is to compute $b \in \mathbb{F}_2^n$ whose existence is stipulated by these theorems.

The definition of the two problems is the following.

PPA-Circuit Chevalley
Input: (C, a), where C is an n-variable PPA-circuit over $\mathbb{F}_2$, and a is a zero of C.
Output: Another zero $b \neq a$ of C.

PPA-Circuit CNSS
Input: (C', a), where C' is an n-variable PPA-circuit over $\mathbb{F}_2$, and $a \in \mathbb{F}_2^n$.
Output: An element $b \in \mathbb{F}_2^n$ satisfying $C = C' \oplus L_a$.

Let us restate here our main theorem.

▶ **Theorem 1** (restated). *The problems* PPA-Circuit CNSS *and* PPA-Circuit Chevalley *are* PPA*-complete.*

Proof. In Proposition 10 below we show that PPA-Circuit CNSS and PPA-Circuit Chevalley are polynomially interreducible. In Theorem 11 in Section 6 we prove that PPA-Circuit CNSS is in PPA, and in Theorem 13 in Section 7 we prove that PPA-Circuit Chevalley is PPA-hard. ◀

We now turn to the proof of the various parts of Theorem 1.

▶ **Proposition 10.** PPA-Circuit CNSS *and* PPA-Circuit Chevalley *are polynomially equivalent.*

Proof. First we reduce PPA-Circuit CNSS to PPA-Circuit Chevalley. Let (C', a) be an instance of PPA-Circuit CNSS, and set $C = C' \oplus L_a$. We can suppose that $C'(a) = 1$, since otherwise we are done. We define the circuit $C'' = C \oplus 1$. Then clearly C'' is a PPA-circuit, and $C''(a) = 0$. The result of the reduction is then the input (C'', a) to PPA-Circuit Chevalley. If the solution to that input is another zero $b \neq a$ of $C''(x)$, then clearly $C(b) = 1$.

The reduction from PPA-Circuit Chevalley to PPA-Circuit CNSS is very similar. Let (C, a) be an instance of PPA-Circuit Chevalley. We set $C' = C \oplus 1$, and $C'' = C' \oplus L_a$. Clearly C' is a PPA-circuit. The result of the reduction is (C', a). If the solution to that input is a satisfying assignment $C''(b) = 1$ then b is a zero of C. Also, $b \neq a$ since $C''(a) = 0$, therefore b is another zero of C. ◀

6 PPA-easiness

▶ **Theorem 11.** PPA-Circuit CNSS *is in* PPA.

Proof. We will give a reduction from PPA-Circuit CNSS to Leaf. Given an input $N = (C', a)$ to PPA-Circuit CNSS, we set $C = C' \oplus L_a$. We construct a graph $G_N = (V_N, E_N)$ by a polynomial time edge recognition algorithm and a polynomial time pairing function ϕ as explained in Section 2.1. The vertices of G_N are $V_N = \mathbb{F}_2^n \cup \mathcal{S}(C)$.

There are two types of edges in E_N, the first type is between an assignment and a parse subcircuit, and the second type is between two maximal parse subcircuits. By definition, the

edge $\{a, S\}$ exists between $a \in \mathbb{F}_2^n$ and $S \in \mathcal{S}(C)$ if $m_S(a) = 1$. Such an edge can be easily recognized since the monomial $m_S(x)$ can be evaluated in linear time in the size of C.

Since C is the disjoint sum of C' and L_a, the maximal parse subcircuits of C are the maximal parse subcircuits of C' extended with the appropriate mark at the output gate, and the unique maximal parse subcircuit of L_a, again extended with the appropriate mark at the output gate. Let us denote the latter parse subcircuit by T. Let μ be a polynomial time computable perfect matching between the maximal parse subcircuits of C', which exists by Lemma 8. By definition, the edge $\{S, S'\}$ exists between $S, S' \in \mathcal{S}(C')$ if both are extensions of maximal parse subcircuits of C', and their restrictions to C' are matched by μ.

Observe that by Proposition 6, a vertex $a \in \mathbb{F}_2^n$ has odd degree if and only if $C(a) = 1$. If S is a maximal parse subcircuit then among the vertices in $\mathbb{F}_2^n$ it is only connected to 1^n. If $S \neq T$, then it has one more neighbor, its matching pair given by μ, and therefore its degree is two. On the other hand, the degree of T is one and therefore it is odd. We can therefore take T as the standard leaf.

We first give the pairing for the vertices in $\mathcal{S}(C)$. We fix $S \in \mathcal{S}(C)$, and let $a \in \mathbb{F}_2^n$ such that $m_S(a) = 1$. If S is not a maximal parse subcircuit then let $i \in [n]$ be the smallest integer such that x_i is not in $m_S(x)$, and let a' be obtained from a by flipping the ith bit. Then by definition $\phi(S, \cdot)$ pairs a with a'. If $S \neq T$ is a maximal parse subcircuit then it has two neighbors: its matching pair S' by μ and 1^n, and $\phi(S, \cdot)$ pairs these two neighbors. For every S, the mapping $\phi(S, \cdot)$ is clearly involutive.

We now turn to the more complicated pairing for the vertices in $\mathbb{F}_2^n$. Observe that this depends only on the edges of the first type, that is edges between an assignment $a \in \mathbb{F}_2^n$ and a parse subcircuit $S \in \mathcal{S}(C)$. These edges can be defined actually for an arbitrary circuit C. Let us denote by $G(C)$ the graph with vertex set $\mathbb{F}_2^n \cup \mathcal{S}(C)$ and with edges of the first type from G_N. First we prove the following lemma about $G(C)$ on induction of the size of C.

▶ **Lemma 12.** *For every n-variable, single-output circuit C, and for every vertex $a \in \mathbb{F}_2^n$ in $G(C)$,*

(a) *if $\deg(a)$ is even then for all $S \in \mathcal{S}(C)$ such that $m_S(a) = 1$, there exists $g \in \mathrm{Dom}(S)$ with $P_g(a) = 0$,*

(b) *if $\deg(a)$ is odd then there exists a unique $S \in \mathcal{S}(C)$ such that $m_S(a) = 1$, and $P_g(a) = 1$ for all $g \in \mathrm{Dom}(S)$.*

Proof. If C consists of a single node, the statement is obviously true. Otherwise we first handle a). When $\deg(a)$ is even then $C(a) = 0$. If the root is a sum gate then we are done since it is in the domain of every parse subcircuit. If the root is a product gate then at least one of its children (say the left without loss of generality) also evaluates to 0, that is $C_\ell(a) = 0$. Let $S \in \mathcal{S}(C)$ be such that $m_S(a) = 1$, then we also have $m_{S_\ell}(a) = 1$. By the inductive hypothesis there exists $g \in \mathrm{Dom}(S_\ell)$ with $P_g(a) = 0$, and since g is also in the domain of S, we are again done.

We now deal with the induction step of b). When $\deg(a)$ is odd then $C(a) = 1$. If the root is a sum gate then one of its children evaluates to 0, and the other one to 1, say $C_\ell(a) = 0$ and $C_r(a) = 1$. By the inductive hypothesis there exists a unique $S' \in \mathcal{S}(C_r)$ such that $m_{S'}(a) = 1$, and $P_g(a) = 1$ for all $g \in \mathrm{Dom}(S')$. On the other hand, if $S \in \mathcal{S}(C)$ such that $m_S(a) = 1$ and the mark of S at the root is ℓ, then $S_\ell \in \mathcal{S}(C_\ell)$ and $m_{S_\ell}(a) = 1$, and by a) there exists $g \in \mathrm{Dom}(S)$ with $P_g(a) = 0$. Therefore the unique S satisfying the hypothesis of the statement is S' extended with the mark r at the root.

To finish the induction step for b), let us suppose now that the root of C is a product gate. Then by the inductive hypothesis there exists a unique $S' \in \mathcal{S}(C_\ell)$ such that $m_{S'}(a) = 1$,

and $P_g(a) = 1$ for all $g \in \mathrm{Dom}(S')$, and similarly there exists a unique $S'' \in \mathcal{S}(C_r)$ such that $m_{S''}(a) = 1$, and $P_g(a) = 1$ for all $g \in \mathrm{Dom}(S'')$. We claim that S' and S'' are compatible, and therefore their union $S = S' \cup S''$ is the unique parse subcircuit of C satisfying the claim. Suppose that it is not the case, that is there exists $g \in \mathrm{Dom}(S') \cap \mathrm{Dom}(S'')$ such that $S'(g) \neq S''(g)$. Since $P_g(a) = 1$, for one of its children, say for g_ℓ, we have $P_{g_\ell}(a) = 0$, contradicting the inductive hypothesis about the parse subcircuit in $\{S', S''\}$ which takes the value ℓ in g. ◀

We give now the pairing $\phi(a, \cdot)$ for $a \in \mathbb{F}_2^n$. If $\deg(a)$ is even then let $S \in \mathcal{S}(C)$ be such that $m_S(a) = 1$. By Lemma 12 there exists a sum gate in the domain of S where P evaluates to 0. Let g be in some topological ordering of the gates of C the first sum gate such that $P_g(a) = 0$, and suppose without loss of generality that $S(g) = \ell$. Let $Z \in \mathcal{S}(C_g)$ be the restriction of S to C_g, and we obviously have $m_Z(a) = m_{Z_\ell}(a) = 1$. We claim that $P_{g_\ell}(a) = P_{g_r}(a) = 1$. Indeed, if $P_{g_\ell}(a) = P_{g_r}(a) = 0$, then by Lemma 12, applied to C_{g_ℓ}, there exists $g' \in \mathrm{Dom}(Z_\ell)$ with $P_{g'}(a) = 0$, which contradicts the choice of g. Therefore again by Lemma 12 there exists a unique $Z'' \in \mathcal{S}(C_{g_r})$ such that $m_{Z''}(a) = 1$, and $P_h(a) = 1$ for all $h \in \mathrm{Dom}(Z'')$. We let $Z' \in \mathcal{S}(C_g)$ be the extension of Z'' with $Z'(g) = r$. Finally we define $\phi(a, S)$ as the parse subcircuit S' obtained from S by exchanging Z with Z', that is $S' = (S \setminus Z) \cup Z'$. It is clear that $m_{S'}(a) = 1$, and $\phi(a, S') = S$.

If $\deg(a)$ is odd then by Lemma 12 there exists a unique parse subcircuit S such that $m_S(a) = 1$, and $P_g(a) = 1$, for all $g \in \mathrm{Dom}(S)$. We set $\phi(a, S) = S$. For all parse subcircuits S such that $P_g(a) = 0$, for some $g \in \mathrm{Dom}(S)$, the construction of $S' = \phi(a, S)$ is identical to the previous case.

The finish the proof, observe that the vertices of odd degree in V_N other than the standard leaf T are the elements $a \in \mathbb{F}_2^n$ such that $C(a) = 1$. Therefore the output of the reduction is a satisfying assignment a for C. ◀

7 PPA-hardness

▶ **Theorem 13.** PPA-Circuit Chevalley *is* PPA*-hard.*

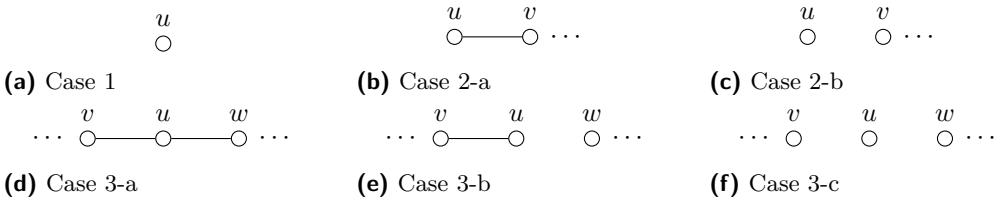
Proof. We will reduce Leaf to PPA-Circuit Chevalley. Let (z, M, ω) be an instance of Leaf, where M defines the graph $G_z = (V_z, E_z)$ with $V_z = \{0,1\}^n$, for some polynomial function n of $|z|$, and ω is the standard leaf in G_z. We know that for every vertex u, $M(z, u)$ is a set of at most two vertices. Composing the standard simulation of polynomial time Turing machines by polynomial size boolean circuits [24] with the obvious simulation of boolean circuits by arithmetic circuits, there exist two n-variables, n-output polynomial size arithmetic circuits D and F with the following properties:

- if $M(z, u) = \emptyset$ or $M(z, u) = \{u\}$ then $D(u) = F(u) = u$,
- if $M(z, u) = \{v\}$ or $M(z, u) = \{v, u\}$ with $v \neq u$ then $D(u) = v$ and $F(u) = u$,
- if $M(z, u) = \{v, w\}$ with $v \neq u \neq w$ then $D(u) = v$ and $F(u) = w$ (or vice versa).

Consider the PPA-composition $C_{D,F}$ of D and F. We claim that for every vertex u, the degree of u in G_z is odd if and only if u is a satisfying assignment for $C_{D,F}$. This is equivalent to saying that the parity of the degree of u is the same as the parity of the satisfied components of $C_{D,F}$. The proof of this claim is straightforward, but somewhat tedious. We distinguish three cases in the proof, depending on the cardinality of $M(z, u) \setminus \{u\}$.

- Case 1: $M(z, u) \setminus \{u\} = \emptyset$. Then u is an isolated vertex, and all six components are satisfied.

- Case 2: $M(z,u) \setminus \{u\} = \{v\}$.
  a) If $u \in M(z,v)$ then the degree of u is one, and $I_5 \diamond F_3 \circ F_4, I_6 \diamond F_5$ and exactly one of the two components $I_2 \diamond F_2 \circ D_2, I_3 \diamond D_3 \circ D_4$ are satisfied.
  b) If $u \notin M(z,v)$ then u is an isolated vertex, and $I_5 \diamond F_3 \circ F_4$ and $I_6 \diamond F_5$ are satisfied.
- Case 3: $M(z,u) \setminus \{u\} = \{v,w\}$.
  a) If $u \in M(z,v) \cap M(z,w)$ then the degree of u is two, and exactly one of the two components $I_2 \diamond F_2 \circ D_2, I_3 \diamond D_3 \circ D_4$ and exactly one of the two components $I_1 \diamond D_1 \circ F_1, I_5 \diamond F_3 \circ F_4$ are satisfied.
  b) If $u \in M(z,v)$ but $u \notin M(z,w)$ and say $D(u) = v$, then exactly one of the two components $I_2 \diamond F_2 \circ D_2, I_3 \diamond D_3 \circ D_4$ is satisfied.
  c) Finally, if $u \notin M(z,v) \cup M(z,w)$ then u is an isolated vertex, and none of the components is satisfied.



(a) Case 1 **(b)** Case 2-a **(c)** Case 2-b

(d) Case 3-a **(e)** Case 3-b **(f)** Case 3-c

Figure 14 The six cases of Theorem 13.

This finishes the proof of the claim. It follows that the number of satisfying assignments for $C_{D,F}$ is equal to the number of leaves in G_z, which is even. The standard leaf ω is a satisfying assignment for $C_{D,F}$, and therefore the output of PPA-Circuit Chevalley is another satisfying assignment, which is another leaf in G_z. ◀

Acknowledgments. Part of this work was performed when A. B., M. S. and S. Y. attended the program "Semidefinite and Matrix Methods for Optimization and Communication" hosted at the Institute for Mathematical Sciences, Singapore. We thank the Institute for the hospitality. G. I. is grateful to the Centre for Quantum Technologies, NUS where part of his research was accomplished.

We are very grateful to several anonymous referees for numerous insightful comments on the paper. We would also like to thank Hervé Fournier, Guillaume Malod and Sylvain Perifel for several helpful conversations.

References

1 J. Aisenberg, M. Bonet and S. Buss. 2–D Tucker is PPA-complete. *ECCC Report* no. 163, 2015.
2 N. Alon. Combinatorial Nullstellensatz. *Combinatorics, Probability and Computing*, 8:1–2, pages 7–29, 1999.
3 N. Alon. Discrete Mathematics: Methods and Challenges. *In Proc. of* 2002 *International Congress of Mathematicians (ICM)*, vol. I, pages 119–135, 2002.
4 P. Beame, S. Cook, J. Edmonds, R. Impagliazzo and T. Pitassi. The relative complexity of NP search problems. *Journal of Computer and System Sciences*, 57(1), pages 3–19, 1998.
5 K. Berman. Parity results on connected f-factors. *Dicrete Math.*, 59, pages 1–8, 1986.
6 J. Bondy and F. Halberstam. Parity theorems for cycles and cycles in graphs. *J. of Graph Theory*, 10, pages 107–115, 1986.

7 K. Cameron and J. Edmonds. Existentially poly-time theorems. *DIMACS Series Discrete Mathematics and Theoretical Computer Science*, 1, pages 83–99, 1990.

8 K. Cameron and J. Edmonds. Some graphic uses of an even number of odd nodes. *Ann. Inst. Fourier* 49, pages 1–13, 1999.

9 X. Chen and X. Deng. On the complexity of 2D discrete fixed point problem. *In Proc. of* 33*rd ICALP*, pages 489–500, 2006.

10 X. Chen and X. Deng. Settling the complexity of two-player Nash equilibrium. *In Proc. of* 47*th FOCS*, pages 261–272, 2006.

11 X. Chen, X. Deng and S.-H. Teng. Settling the complexity of computing two-player Nash equilibria. *J. ACM*, 56(3), pages 1–57, 2009.

12 C. Chevalley. Démonstration d'une hypothèse de M. Artin. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, 11, pages 73–75, 1936.

13 X. Deng, J. Edmonds, Z. Feng, Z. Liu, Q. Qi and Z. Xu. Understanding PPA-completeness. *in Proc. of* 31*st CCC*, pages 23:1–23:25, 2016.

14 M. Grigni. A Sperner lemma complete for PPA. *Inform. Process. Lett.*, 77(5-6), pages 255–259, 2001.

15 E. Jeřábek. Integer factoring and modular square roots. *Journal of Computer and System Sciences*, 82, no. 2, pages 380–394, 2016.

16 Mark Jerrum and Marc Snir. Some Exact Complexity Results for Straight-Line Computations over Semirings. *J. ACM*, 29, no. 3, pages 874–897, 1982.

17 K. Friedl, G. Ivanyos, M. Santha and Y. Verhoeven. Locally 2-dimensional Sperner problem complete for the Polynomial Parity Argument classes. *In Proc.* 6*th CIAC*, pages 380–391, 2006.

18 S. Kintali. A compendium of PPAD-complete problems.

19 G. Malod. Polynômes et coefficients. *Doctoral Thesis*, Université Claude Bernard, Lyon 1, 2003.

20 G. Malod and N. Portier. Characterizing Valiant's algebraic complexity classes. *Journal of Complexity*, 24, pages 16–38, 2008.

21 N. Megiddo and C. Papadimitriou. On total functions, existence theorems and computational complexity. *Theoret. Comput. Sci.*, 81, pages 317–324, 1991.

22 C. Papadimitriou. On graph-theoretic lemmata and complexity classes. *In Proc. of* 31*st FOCS*, pages 794–801, 1990.

23 C. Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *J. Comput. System Sci.*, 48(3), pages 498–532, 1994.

24 M. Sipser. Introduction to the Theory of Computation. PWS Publishing Company, 1997.

25 S. Toida. Properties of an Euler graph. *J. Franklin Institute*, 95, pages 343–345, 1973.

26 Leslie Valiant. Completeness for parity problems. In *International Computing and Combinatorics Conference*, pages 1–8. Springer, 2005.

27 L. Varga. Combinatorial Nullstellensatz modulo prime powers and the Parity Argument. *Electr. J. Comb.*, 21(4), P4.44, 2014.

28 E. Warning. Bemerkung zur vorstehenden Arbeit von Herrn Chevalley. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, 11, pages 76–83, 1936.

29 D. West. Pairs of adjacent Hamiltonian circuits with small intersection. *Studies of Applied Math.*, 59, pages 245–248, 1978.

An Exponential Lower Bound for Homogeneous Depth-5 Circuits over Finite Fields*

Mrinal Kumar[1] **and Ramprasad Saptharishi**[2]

1 **Rutgers University, New Brunswick, NJ, USA**
mrinalkumar08@gmail.com
2 **Tata Institute of Fundamental Research, Mumbai, India**
ramprasad@tifr.res.in

Abstract

In this paper, we show exponential lower bounds for the class of homogeneous depth-5 circuits over all small finite fields. More formally, we show that there is an explicit family $\{P_d\}$ of polynomials in VNP, where P_d is of degree d in $n = d^{O(1)}$ variables, such that over all finite fields $GF(q)$, any homogeneous depth-5 circuit which computes P_d must have size at least $\exp(\Omega_q(\sqrt{d}))$.

To the best of our knowledge, this is the first super-polynomial lower bound for this class for any non-binary field.

Our proof builds up on the ideas developed on the way to proving lower bounds for homogeneous depth-4 circuits [Gupta et al., Fournier et al., Kayal et al., Kumar-Saraf] and for non-homogeneous depth-3 circuits over finite fields [Grigoriev-Karpinski, Grigoriev-Razborov]. Our key insight is to look at the space of shifted partial derivatives of a polynomial as a space of functions from $GF(q)^n$ to $GF(q)$ as opposed to looking at them as a space of formal polynomials and builds over a tighter analysis of the lower bound of Kumar and Saraf [Kumar-Saraf].

1998 ACM Subject Classification I.1.1 Expressions and Their Representation

Keywords and phrases arithmetic circuits, lower bounds, separations, depth reduction

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.31

1 Introduction

Arithmetic circuits are the most natural model to study computations of multivariate polynomials. These are directed acyclic graphs, with a unique sink node called the root or output gate, internal nodes are labeled by addition and multiplication gates[1], and leaves are labeled by variables or constants from the underlying field. The field of arithmetic circuit complexity aims at understanding the hardness of multivariate polynomials in terms of the size of the smallest arithmetic circuit computing it. One of the most important questions in this field of study is to show that there are families of explicit *low-degree*[2] polynomials that require arithmetic circuits of super-polynomial size (in terms of n, the number of variables). It is widely believed that the symbolic $n \times n$ permanent, often denoted by Perm_n, requires circuits of size $\exp(\Omega(n))$ but, as of now, we do not even have a $\Omega(n^2)$ lower bound for any explicit polynomial.

* The first author's research supported in part by a Simons Graduate Fellowship. The second author's research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement number 257575.

[1] Throughout this paper, we consider circuits as having gates of unbounded fan-in.

[2] Where the degree is bounded by a polynomial function in the number of variables.

32nd Computational Complexity Conference (CCC 2017).
Editor: Ryan O'Donnell; Article No. 31; pp. 31:1–31:30

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

Depth Reductions

In the absence of much progress on the question of lower bounds for general arithmetic circuits, a natural question to ask is if we can prove good lower bounds for nontrivial restricted classes of circuits. One particular class of circuits which have been widely studied with this aim are the class of bounded depth[3] arithmetic circuits. It turns out that this is not just an attempt to study a simpler model, but there is a formal connection between lower bounds for bounded depth circuits and lower bounds for general circuits. A sequence of structural results, often referred to as *depth reduction* results, show that strong enough lower bounds for bounded depth circuits implies lower bounds for general arithmetic circuits.

The first depth reduction for arithmetic circuits was by Hyafil [10] who showed that any polynomial computed by a polynomial sized arithmetic circuit can be equivalently computed by a circuit of depth $O(\log d)$ and *quasi-polynomial* size. This was improved by Valiant, Skyum, Berkowitz and Rackoff [26], who showed that any n-variate degree d polynomial that can be computed by a circuit of size $(nd)^{O(1)}$ can be equivalently computed by a circuit of depth $O(\log d)$ and size $(nd)^{O(1)}$. Thus, proving super-polynomial lower bounds for $O(\log d)$ depth circuits is sufficient to prove super-polynomial lower bounds for general arithmetic circuits. Agrawal and Vinay [2] further strengthened this to obtain a depth reduction to depth-4 circuits by showing that any n-variate degree d polynomial that can be computed by a $2^{o(n)}$ sized circuit can be equivalently computed by *homogeneous*[4] depth-4 circuit of size $2^{o(n)}$. Their result was strengthened by Koiran [16] and Tavenas [25] to show that any circuit of size s that computes an n-variate degree d polynomial can be computed by a homogeneous depth-4 circuit of size $s^{O(\sqrt{d})}$, and in fact the resulting depth-4 circuits have all multiplication fan-ins bounded by $O(\sqrt{d})$. These results hold over all fields.

Over any field of characteristic zero, Gupta, Kamath, Kayal and Saptharishi [8] showed that any n-variate degree d polynomial computed by a size s circuit can be equivalently computed by a non-homogeneous depth-3 circuit of size $s^{O(\sqrt{d})}$. Thus, these results formally show that proving good enough lower bounds on circuits of bounded depth is sufficient for proving lower bounds for general circuits.

Lower bounds for depth-3 and depth-4 circuits

Nisan and Wigderson [20] proved an $\exp(\Omega(n))$ lower bound for any *homogeneous* depth-3 circuits computing the symbolic $n \times n$ determinant Det_n by studying dimension of the partial derivatives of Det_n as polynomials. Grigoriev and Karpinski [6] and Grigoriev and Razborov [7] extended this to prove an $\exp(\Omega(n))$ lower bound for non-homogeneous depth-3 circuit computing Det_n over any fixed finite field $\mathbb{F}_q$. Chillara and Mukhopadhyay [4] extended this to give an $\exp(\Omega_q(d \log n))$ lower bound for non-homogeneous depth-3 circuits computing an n-variate degree d polynomial in VP. It is worth noting that there is no generic method known to convert a boolean lower bound for $\mathsf{AC}^0[\mathrm{mod}q]$ to lower bounds for arithmetic circuits over $\mathbb{F}_q$ (discussed in more detail in Section 3.1).

The proofs of [6, 4] also studied the dimension of partial derivatives of polynomial, but unlike the proof in [20], they looked at partial derivatives as functions from $\mathbb{F}_q^n \to \mathbb{F}_q$. The proofs in [6], [7] and [4] strongly rely on the fact that we are working over small finite

[3] A depth k arithmetic circuit consists of k layers of alternating sum and multiplication gates with the output being computed by a sum gate.

[4] Which means that all intermediate computations are homogeneous polynomials. Hence the degree of any intermediate computation is bounded by the degree of the output polynomial.

fields, and completely break down over larger fields or fields of large characteristic. Over fields of characteristic zero and over algebraic closure of finite fields, the question of proving super-polynomial lower bounds for non-homogeneous depth three circuits continues to remain wide open.

Even though we had exponential lower bounds for homogeneous depth-3 circuits, the question of proving super-polynomial lower bounds for homogeneous depth-4 circuits remained open for more than a decade. In 2012, Kayal [12] introduced the notion of *shifted partial derivatives*, which is a generalization of the well-known notion of partial derivatives. Shifted partial derivatives have been very influential in a plethora of lower bounds for depth-4 circuits in the past few years. Gupta et. al. [9] used this measure to prove an $\exp(\Omega(\sqrt{n}))$ lower bound for the size of homogeneous depth-4 circuits with multiplication fan-ins bounded by $O(\sqrt{n})$. Subsequently, lower bounds of $\exp(\Omega(\sqrt{d}\log n))$ were proved for other n-variate degree d polynomials computed by almost the same circuit class [15, 5, 18]. (It is worth noting that getting a lower bound of $\exp(\omega(\sqrt{d}\log n))$ would have implied a super-polynomial lower bound for general circuits!) Using a more delicate variant called *projected shifted partials*, Kayal et. al. [13] and Kumar and Saraf [19] proved lower bounds of $\exp(\Omega(\sqrt{d}\log n))$ for homogeneous depth-4 circuits (without any fan-in restrictions) via two very different analyses. The former was an analytic approach and works only over characteristic zero fields, whereas the latter was purely combinatorial and works over any field. These techniques have also been applied to yield lower bounds for non-homogeneous depth-3 circuits with bounded bottom fan-in [14] and homogeneous depth-5 circuits with bounded bottom fan-in [3]. A continuous updated survey [23] contains expositions of many of the lower bounds and depth reduction results listed above.

The results in [19] in fact show that the reduction from general arithmetic circuits to depth-4 circuits with support $O(\sqrt{d})$ cannot be improved, as they give an example of a polynomial in VP for which any depth-4 circuits of support $O(\sqrt{d})$ must be of size $n^{\Omega(\sqrt{d})}$. Further, with the current upper-bounds for the projected shifted partials on such depth-4 circuits, the best we can hope to prove using this measure is an $n^{\Omega(\sqrt{d})}$ lower bound. Hence, it might be insufficient for general arithmetic circuits lower bounds but it could well be the case that we might be able to prove stronger lower bounds for constant depth arithmetic circuits, or arithmetic formulas by variants of this family of measures.

Hence, as a start, the problem of proving lower bounds for homogeneous depth five circuits, seems like the next natural question to explore. This already seems to introduce new challenges as the proofs of lower bounds for homogeneous depth-4 circuits seem to break down for homogeneous depth-5 circuits. In this paper, we pursue this line of enquiry, and prove exponential lower bounds for homogeneous depth-5 circuits over small finite fields. Before stating our results, we first discuss prior results on this question, and the challenges involved in extending the proofs of lower bounds for homogeneous depth four circuits, in the next section.

Lower bounds for depth-5 circuits

Prior to this work, the only known lower bounds for depth-5 circuits that we are aware of are the results of Raz [21], which show super-linear lower bounds for bounded depth circuits over large enough fields, the results of Kalorkoti [11] which show quadratic lower bounds for arithmetic formulas and the results of Kayal and Saha [14] and of Bera and Chakrabarti [3] which show exponential lower bounds for homogeneous depth-5 circuits if the bottom fan-in is bounded.

Given that we have lower bounds for homogeneous depth-4 circuits, it seems natural to try and apply these techniques to prove lower bounds for homogeneous depth-5 circuits.

Unfortunately, the obvious attempts to generalize the proofs in [13, 19] seem to fail for homogeneous depth-5 circuits. We now elaborate on this.

On extending the depth-4 lower bound proofs to depth-5 circuits: To understand these issues, we first need a birds-eye view of the major steps in the proofs of lower bounds for depth-4 circuits [13, 19]. These proofs have two major components.

- **Reduction to depth-4 circuits with bounded bottom support:** In the first step, the circuit C and the polynomial are hit with a random restriction, in which each variable is kept alive independently with some small probability p. The observation is that a bottom level product gate in C of support (the number of distinct variable inputs) at least s survives with probability at most p^s. Therefore, the probability that some bottom product of support at least s in C survives is at most $\text{Size}(C) \cdot p^s$. Now, if the size of C is small (say $\epsilon \cdot 1/p^s$), then this probability is quite small, so with a high probability C reduces to a homogeneous depth-4 circuit with bounded bottom support.
- **Lower bounds for depth-4 circuits with bounded bottom support:** The goal in the second step is to show that the polynomial obtained after random restrictions still remains hard for homogeneous depth-4 circuits with bottom support at most s.

The key point in step 1 is that if $\text{Size}(C)$ is not too large, then we can assume that with a high probability over the random restrictions, all the high support product gates are set to 0. This is where things are not quite the same for depth-5 circuits. When we express a homogeneous depth-5 circuit as a homogeneous depth-4 circuit by expanding the product of linear forms at level four, we might increase the number of monomials a lot (potentially to all possible monomials). Now, the random restriction step no longer works and we do not have a reduction to homogeneous depth-4 circuits with bounded bottom support. If the bottom fan-in of C is bounded, then this strategy does indeed generalize. Kayal and Saha [14] and Bera and Chakrabarti [3] show exponential lower bounds for such cases.

It is not clear to us how fundamental this obstruction is, but our key insight is a strategy for proving lower bounds for homogeneous depth-4 circuits that avoids the random restriction step. Morally speaking, we *do* proceed by a 'reduction' from a depth-5 circuit to a depth-4 circuit, but the meaning of a 'reduction' here is more subtle and largely remains implicit.

Our Contribution

We give an exponential lower bound for homogeneous depth-5 circuits over any fixed finite field $\mathbb{F}_q$. To the best of our understanding, this is the first such lower bound for depth-5 circuits over any field apart from $\mathbb{F}_2$[5]. Stated precisely, we prove the following theorem.

▶ **Theorem 1.** *There is an explicit family of polynomials* $\{P_d : d \in \mathbb{N}\}$*, with* $\mathsf{Deg}(P_d) = d$*, in the class* VNP *such that for any finite field* $\mathbb{F}_q$*, any homogeneous depth-5 circuit computing* P_d *must have size* $\exp(\Omega_q(\sqrt{d}))$*.*

The polynomial P_d is from the Nisan-Wigderson family of polynomials (introduced by [15], Definition 3) with carefully chosen parameters.

Our proof also extends to non-homogeneous depth-5 circuits where the layer of multiplication gates closer to the output have fan-in bounded by $O(\sqrt{d})$ (with no restriction on the fan-in of the other multiplication layer).

[5] For $\mathbb{F}_2$, exponential lower bounds easily follow from the lower bounds of Razborov [22].

▶ **Theorem 2.** *There is an explicit family of polynomials* $\{P_d : d \in \mathbb{N}\}$*, with* $\mathsf{Deg}(P_d) = d$*, in the class* VNP *such that for any finite field* $\mathbb{F}_q$*, any* $\Sigma\Pi^{[O(\sqrt{d})]}\Sigma\Pi\Sigma$ *circuit computing* P_d *must have size* $\exp(\Omega_q(\sqrt{d}))$*.*

It is worth mentioning that for characteristic zero fields, it suffices to prove a lower bound of $\exp(\omega(d^{1/3}\log d))$ for an explicit polynomial computed by such $\Sigma\Pi^{[O(\sqrt{d})]}\Sigma\Pi\Sigma$ circuits to separate VP from VNP (by combining the depth reductions of [2, 16, 25] and [8]). We elaborate on this in Section 3.5. Such a phenomenon also happens for non-homogeneous depth three circuits, where over finite fields, we know quite strong lower bounds while much weaker ones would imply $\mathsf{VNP} \neq \mathsf{VP}$ over fields of characteristic zero.

The key technical ingredient of our proof is to look at the space of shifted partial derivatives and the projected shifted partial derivatives of a polynomial. We study them as a space of functions from $\mathbb{F}_q^n \to \mathbb{F}_q$ as opposed to as a space of formal polynomials, as has been the case for the results obtained so far. This perspective allows us the freedom to confine our attention to the evaluations of the shifted partial derivatives of a polynomial on certain well chosen subsets of $\mathbb{F}_q^n$, and this turns out to be critical to our cause. This leads to a new family of complexity measures which could have applications to other lower bound questions as well. Our proof also involves a tighter analysis of the lower bound of Kumar and Saraf [19] (for homogeneous depth-4 circuits) which may be interesting in its own right.

We now give an overview of our proof.

2 An overview of the proof

The proof would consist of the following main steps:

1. Define a function $\Gamma : \mathbb{F}_q[\mathbf{x}] \to \mathbb{N}$. Intuitively, we think of $\Gamma(P)$ to be a measure of the *complexity* of P.
2. For all homogeneous depth-5 circuits C of size at most $\exp(\delta\sqrt{d})$, prove an upper bound on $\Gamma(C)$.
3. For the target hard polynomial P, show that $\Gamma(P)$ is much larger than the upper bound proved in step 2.

The complexity measure: At a high level, the proofs of lower bounds in [20, 9, 15, 5, 18, 13, 19] associate a linear space polynomials to every polynomial in $\mathbb{F}_q[\mathbf{x}]$ and use the dimension of this space over $\mathbb{F}_q$ as a measure of complexity of the polynomial. The mapping from polynomials to linear space of polynomials undergoes subtle changes as we go from the proof of lower bounds for homogeneous depth-3 circuits [20] to lower bounds for homogeneous depth-4 circuits [13, 19].

In this paper, we follow this outline and associate to every polynomial, the space of its shifted partial derivatives as defined in [9]. However, instead of working with this space of polynomials as it is, we study their evaluation vectors over a subset of $\mathbb{F}_q^n$ (similar to [6, 7], where they worked with partial derivatives of a polynomial). The key gain that we have from this change in outlook is that as evaluation vectors, we can choose to confine our attention to evaluations on certain properly chosen subsets of $\mathbb{F}_q^n$. For formal polynomials, it is not clear what should be the correct analog of this approximation. The necessity and the utility of this will be more clear as we go along.

High rank products of linear forms: Consider a polynomial Q which is a product of τ linearly independent (non-constant) linear forms $L_1, L_2, \ldots, L_\tau$.

$$Q = \prod_{i=1}^{\tau} L_i \,.$$

It is not hard to see that

$$\Pr_{\mathbf{a}\in\mathbb{F}_q^n}[Q(\mathbf{a}) \neq 0] \leq \left(1 - \frac{1}{q}\right)^{\tau} .$$

In other words, products of linear forms of rank τ vanish on all but a $o(1)$ fraction of the entire space if $\tau = \omega(1)$. If the size of a depth-5 circuit is not too large as a function of τ (say, at most $\exp(\delta\tau)$ for a small enough $\delta > 0$), then by a union bound, all the products of rank at least τ at the fourth level vanish everywhere apart from a $o(1)$ fraction of the points in $\mathbb{F}_q^n$.

In summary, we just argued that a depth-5 circuit C over $\mathbb{F}_q$ of size at most $\exp(\delta\tau)$ can be approximated by a sub-circuit C' of C which is obtained from C by dropping all products of linear forms of rank at least τ from the bottom level.

Low rank products of linear forms: A second simple observation (Lemma 8) shows that for every product of linear forms of rank at most τ, there is a polynomial of degree at most $(q-1)\tau$, such that they agree at all points in $\mathbb{F}_q^n$. Thus, the circuit C' is equal, as a function from $\mathbb{F}_q^n \to \mathbb{F}_q$ to a depth-4 circuit C'' of bottom fan-in at most $(q-1)\tau$. Moreover, the formal degree and the top fan-in of C'' are upper bounded by the formal degree and top fan-in of C, respectively.

Putting things together This implies that for every homogeneous depth-5 circuit C computing a polynomial of degree d of size at most $\exp(\delta\tau)$ for some τ, there exists a depth-4 circuit C'' of formal degree at most d and top fan-in at most $\exp(\delta\tau)$ such that

$$\Pr_{\mathbf{a}\in\mathbb{F}_q^n}[C(\mathbf{a}) \neq C''(\mathbf{a})] \leq o(1).$$

Therefore, a polynomial P which can be computed by C can be *approximated* by C'' in the point-wise sense. Since we know lower bounds on the top fan-in of homogeneous (and low formal degree) depth-4 circuits with bounded bottom fan-in [9, 15], it seems that we only have a small way to go. Unfortunately, we do not quite know how to make this idea work. The key technical obstacle here is that it seems to be hard to say much about the partial derivatives of C by looking at partial derivatives of C''. As a pathological case, the polynomial $\prod_{i\in[n]} x_i$ has a partial derivative span of size 2^n but is well approximated by the 0 polynomial over $\mathbb{F}_2$.

If we had started with a depth-3 circuit instead of a depth-5 circuit, then such a strategy is indeed known to work [7]. Observe that in this case it is enough to show that that there is an explicit polynomial which cannot be approximated well by a low degree polynomial over $\mathbb{F}_q$. In [7], the authors show this by an adaptation of a similar result of Smolenksy [24] over $\mathbb{F}_2$.

A strengthening of the strategy: The key additional observation that helps us make things work is the fact that not only do high rank product gates at level four of C vanish almost everywhere on $\mathbb{F}_q^n$, but they vanish with a high multiplicity. As we show in Corollary 11, if

the size of C is not *too large*, all the product gates of rank at least τ vanish with a multiplicity $\Omega(\tau)$ at $1 - o(1)$ fraction of points on $\mathbb{F}_q^n$[6].

Therefore, not only can C agree with C' almost everywhere, all the partial derivatives of C of order at most $k = \Omega(\tau)$ agree with all the partial derivatives of C' almost everywhere. This already hints at the fact that if we are to take advantage of this then we should be looking at the evaluation of these derivatives, since our only guarantee is about their evaluations.

In [6], exponential lower bounds were proved for non-homogeneous depth-3 circuits using a very similar strategy. However, adapting the method for shifted partials and projected shifted partials seems to be a challenge.

In Section 5, we show that the dimension of the span of evaluation vectors of shifted partial derivatives of C, when restricted to a properly chosen subset S of $\mathbb{F}_q^n$, is small if the size of the circuit C we started with was small.

As a final step of the proof, we show that with respect to this complexity measure, our target hard polynomial (from the so-called *Nisan-Wigderson* family, defined below) has a large complexity.

▶ **Definition 3** (Nisan-Wigderson polynomial families)**.** Let d, m, e be arbitrary parameters with m being a power of a prime, and $d, e \leq m$. Since m is a power of a prime, let us identify the set $[m]$ with the field $\mathbb{F}_m$ of m elements. Note that since $d \leq m$, we have that $[d] \subseteq \mathbb{F}_m$. The Nisan-Wigderson polynomial with parameters d, m, e, denoted by $\mathsf{NW}_{d,m,e}$ is defined as

$$\mathsf{NW}_{d,m,e}(\mathbf{x}) \quad = \quad \sum_{\substack{p(t) \in \mathbb{F}_m[t] \\ \mathsf{Deg}(p) < e}} x_{1,p(1)} \dots x_{d,p(d)}$$

That is, for every univariate polynomial $p(t) \in \mathbb{F}_m[t]$ of degree less than e, we add one monomial that encodes the 'graph' of p on the points $[d]$. This is a homogeneous, multilinear polynomial of degree d over dm variables with exactly m^e monomials.

This step of the proof builds on a tighter analysis of the lower bound on the dimension of the span of *projected shifted partial derivatives* of the Nisan-Wigderson polynomials in [19]. We show that if the set S is carefully chosen, then we do not incur much loss in the dimension by going from looking at shifted partial derivatives as formal polynomials to looking at them as functions over a small subset of the finite field. We provide the details in Section 6.

One important technicality is the dependency between various parameters involved. For our proof, the choice of k would be $O_q(\tau)$ and would depend on q. The lower bound of [19] would then choose specific parameters for the $\mathsf{NW}_{d,m,e}$. This would mean that for every q, we get a *different* polynomial for which we show a lower bound. We remedy the order of quantifiers and start by fixing specific parameters for $\mathsf{NW}_{d,m,e}$. Then, depending on q, we choose a restriction of $\mathsf{NW}_{d,m,e}$ that would be identical to $\mathsf{NW}_{d',m,e}$ by setting some variables to 0/1. We then apply the [19] argument for this restriction to obtain our lower bound for $\mathsf{NW}_{d',m,e}$ which also yields a lower bound for $\mathsf{NW}_{d,m,e}$. The details are in Section 7.1.

3 Some discussion and open problems

3.1 Connections between arithmetic circuits over $\mathbb{F}_q$ and $\mathsf{AC}^0[\mathrm{mod}\, q]$

Although constant depth arithmetic circuits over $\mathbb{F}_q$ appear to be similar to the class $\mathsf{AC}^0[\mathrm{mod} q]$, they are surprisingly very different with respect to functions computed by them.

[6] In the rest of this discussion, we will think of τ as $\Theta(\sqrt{d})$.

A striking example, due to Agrawal, Allender and Datta [1], is that arithmetic circuits over $\mathbb{F}_3$ can "compute" both the Mod3 function, as well as the Mod2 function via

$$\text{Mod2}(x_1, \ldots, x_n) \quad = \quad \left(2 + \prod_{i=1}^{n}(1 + x_i)\right)^2.$$

However, it is true that functions computed by arithmetic circuits over $\mathbb{F}_{p^k}$ have strong connections with $\mathsf{AC}^0[\text{mod}p(p^k - 1)]$ but unless we are working over $\mathbb{F}_2$ it seems difficult to lift a lower bound for $\text{AC}^0[\text{mod}p]$ to arithmetic circuits over $\mathbb{F}_p$. For more on this, see [1].

The only exception we know of is the result of Grigoriev and Razborov [7] where they lift Smolensky's [24] lower bound for $\text{AC}^0[\text{mod}p]$ to depth-3 arithmetic circuits over $\mathbb{F}_p$, and this crucially uses the fact that depth-3 arithmetic circuits can be point-wise approximated by a "sparse polynomial". But in general, constant depth arithmetic circuits over $\mathbb{F}_p$ and Boolean circuits in $\text{AC}^0[\text{mod}p]$ seem to be two very different classes.

3.2 Lower bounds for iterated matrix multiplication

Given the results in this paper, one might wonder if the lower bounds in this paper work for a polynomial in VP. One natural candidate polynomial for which one might hope to show such a lower bound would be the iterated matrix multiplication polynomial (IMM). It was shown in [19] that IMM has a large complexity with respect to the measure of projected shifted partial derivatives. Unfortunately, the bounds in [19] only show that the dimension of the space of projected shifted partial derivatives of the IMM (degree d in $d^{O(1)}$ variables) are a factor $\exp(\delta\sqrt{d}\log d)$ close to the maximum possible value for some constant δ. This slack seems to be insufficient for our proofs in this paper to work as in the proof of Lemma 24, we would have to rely on the fact that for the polynomial NW, the projected shifted partials complexity was at most a quasi-polynomial factor away from the largest possible.

3.3 Finer separations for bounded depth circuits?

In [18], it was shown that homogeneous depth-4 circuits are exponentially more powerful than homogeneous depth-4 circuits with bounded bottom fan-in. A natural question to ask is whether such separations can be shown between homogeneous depth-4 circuit and homogeneous depth-5 circuits. One of the first strategies to attempt for this question would be to try and show that there is a homogeneous depth-5 circuit such that its projected shifted partial derivative complexity is quite large. The results in this paper show that the measure can not to be too close to the largest possible value, in particular it needs to be at least a factor $\exp(\Omega(\sqrt{d}))$ away from the largest possible value. If this is bound is tight, then such a separation between homogeneous depth-5 circuits and homogeneous depth-4 circuits can still be shown using projected shifted partial derivatives. However, it is not clear if this is the case. As mentioned before, even the known lower bounds on the dimension of the projected shifted partials for the IMM seem a factor $\exp(\Omega(\sqrt{d}\log d))$ away from the largest possible value.

In a recent result [17], using a different measure (called dimension of shifted projected partials, first used by [13]) such a separation between homogeneous depth-4 and homogeneous depth-5 circuits was shown in the regime when $d = O(\log^2 n)$. Extending this for other regimes of d continues to remain open.

3.4 The tightness of the results and relevance to VP vs. VNP

For homogeneous depth-4 circuits, we know $\exp(\Omega(\sqrt{d}\log d))$ lower bounds [13, 19] and any asymptotic improvement in the exponent would imply general arithmetic circuit lower bounds. In this sense, the lower bounds for homogeneous depth-4 circuits are tight, over all fields. It is natural to ask, if the bounds in this paper are tight in this sense? The answer to this question is far from obvious to us. In particular, it is not clear if we can use computational advantage of having linear forms at the bottom level of the circuit to get a better depth reduction from VP to homogeneous depth-5 circuits, when compared to depth reduction to homogeneous depth-4 circuits.

3.5 Lower bounds over fields of characteristic zero?

One might wonder if the techniques in this paper could be potentially adapted to work for depth-5 circuits over fields of characteristic zero. As in the work of Grigoriev and Karpinski [6], our proof (Lemma 13 in particular) strongly relies on the fact that we are working over a fixed finite field, so it clearly seems hard to generalize over large fields (even when the characteristic is small). In addition to this obvious technical obstruction to generalizing the proof in this paper, there seems to be another reason why the proof strategy in this paper could be hard to replicate over fields of characteristic zero, namely, an analog of Theorem 2 over fields of characteristic zero would imply that VP $\neq$ VNP. The reason is that over characteristic zero fields, one can obtain better depth reductions to non-homogeneous depth-5 circuits by combining [2, 16, 25] with [8]. Although this is reasonably well known, we give a formal proof here for completeness.

The following lemma is a simple generalization of the proof of depth reduction to depth-4 circuits by Tavenas [25].

▶ **Lemma 4** (Depth reduction to homogeneous depth six circuits). *Let P be a polynomial of degree d in* $\text{poly}(d)$ *variables which can be computed by an arithmetic circuit C of size* $\text{poly}(d)$*. Then, there is a homogeneous depth-6 circuit C' which computes P and satisfies*

- *Size(C)* $\leq \exp(O(d^{1/3}\log d))$*, and*
- *The fan-in of all the product gates in C' is bounded by $O(d^{1/3})$.*

Now, we start with the circuit C' as guaranteed by the lemma above, and for each of the product gates at the second level, look at its inputs. Each such input is a $\Sigma\Pi^{O(d^{1/3})}\Sigma\Pi^{O(d^{1/3})}$ circuit (depth-4 circuit with all product fan-ins being at most $O(d^{1/3})$) of size at most $\exp(O(d^{1/3}\log d))$. We now apply the depth reduction to depth-3 by Gupta et al. [8] to each one of these depth-4 circuits. As a result, each of these depth-4 circuits get reduced to a depth-3 circuit, with at most a factor of $\exp(O(d^{1/3}))$ blow up in size. Plugging these depth-3 circuits back into C', we obtain a depth-5 circuit C'' such that

- Size(C) $\leq \exp(O(d^{1/3}\log d))$, and
- The fan-in of all the product gates at level two of C'' is bounded by $O(d^{1/3})$.

Recall that the depth reduction in [8] only works over fields of characteristic zero. This yields the following depth reduction to non-homogeneous depth-5 circuits.

▶ **Lemma 5** (Depth reduction to non-homogeneous depth-5 circuits). *Let $\mathbb{F}$ be a field of characteristic zero. Let P be a polynomial of degree d in* $\text{poly}(d)$ *variables over $\mathbb{F}$ which can be computed by an arithmetic circuit C of size* $\text{poly}(d)$*. Then, there is a depth-5 circuit C'' which computes P and satisfies*

- *Size(C)* $\leq \exp(O(d^{1/3}\log d))$*, and*
- *The fan-in of all the product gates at level two of C' is bounded by $O(d^{1/3})$.*

Now, observe that an analogue of Theorem 2 over fields of characteristic zero, would imply an $\exp(\Omega(d^{1/2}))$ lower bound for the depth-5 circuits obtained in Lemma 5, and hence imply $\mathsf{VP} \neq \mathsf{VNP}$.

4 Notation

- Throughout the paper, we shall use bold-face letters such as $\mathbf{x}$ to denote a set $\{x_1, \ldots, x_n\}$. Most of the times, the size of this set would be clear from context. We shall also abuse this notation to use $\mathbf{x}^{\mathbf{e}}$ to refer to the monomial $x_1^{e_1} \cdots x_n^{e_n}$.
- For an integer $m > 0$, we shall use $[m]$ to denote the set $\{1, \ldots, m\}$.
- We shall use the short-hand $\partial_{\mathbf{x}^{\mathbf{e}}}(P)$ to denote

$$\frac{\partial^{e_1}}{\partial x_1^{e_1}}\left(\frac{\partial^{e_2}}{\partial x_2^{e_2}}\left(\cdots(P)\cdots\right)\right).$$

- For a set of polynomials $\mathcal{P}$ shall use $\partial^{=k}\mathcal{P}$ to denote the set of all k-th order partial derivatives of polynomials in $\mathcal{P}$, and $\partial^{\leq k}\mathcal{P}$ similarly.
  Also, $\mathbf{x}^{=\ell}\mathcal{P}$ shall refer to the set of polynomials of the form $\mathbf{x}^{\mathbf{e}} \cdot P$ where $\mathsf{Deg}(\mathbf{x}^{\mathbf{e}}) = \ell$ and $P \in \mathcal{P}$. Similarly $\mathbf{x}^{\leq \ell}\mathcal{P}$.
- For a polynomial $P \in \mathbb{F}_q[\mathbf{x}]$ and for a set $S \subseteq \mathbb{F}_q^n$, we shall denote by $\mathsf{Eval}_S(P)$ the vector of the evaluation of P on points in S (in some natural predefined order like say the lexicographic order). For a set of polynomials $\mathcal{P}$, $\mathsf{Eval}_S(\mathcal{P})$ denotes the set $\{\mathsf{Eval}_S(P) : P \in \mathcal{P}\}$. For a set of vectors V, their span over $\mathbb{F}_q$ will be denoted by $\mathsf{Span}(V)$ and their dimension by $\mathsf{Dim}(V)$.
- We shall use $\mathcal{H}$ to denote the set $\{0,1\}^n \subset \mathbb{F}_q^n$.
- A polynomial of the form $P = \alpha_0 + \sum_{i=1}^n \alpha_i x_i$, where each $\alpha_j \in \mathbb{F}$ is referred to as an affine form. If $\alpha_0 = 0$, then P is referred to as a linear form or a linear polynomial over $\mathbb{F}$.

The complexity measure

We now define the complexity measure that we shall be using to prove the lower bound. The measure will depend on a carefully chosen set $S \subset \mathbb{F}_q^n$.

▶ **Definition 6** (The complexity measure)**.** Let k, ℓ be some parameters and let $S \subset \mathbb{F}_q^n$. For any polynomial P, define $\Gamma_{k,\ell,S}(P)$ as

$$\Gamma_{k,\ell,S}(P) \quad := \quad \mathsf{Dim}\left\{\mathsf{Eval}_S\left(\mathbf{x}^{=\ell}\partial^{=k}(P)\right)\right\}.$$

5 Complexity measure on a depth-5 circuit

A depth-5 circuit computes a polynomial of the form

$$C \quad = \quad \sum_a \prod_b \sum_c \prod_d L_{abcd} \tag{5.1}$$

where each L_{abcd} are linear polynomials.

▶ **Definition 7** (Terms of a circuit, and rank)**.** For a depth-5 circuit such as (5.1), we shall denote by $\mathrm{Terms}(C)$ the set

$$\mathrm{Terms}(C) \quad := \quad \left\{\prod_d L_{abcd}\right\}_{a,b,c}$$

which are all products of linear polynomials computed by the bottommost product gates.

For any term $T = \prod_d L_d$, define $\mathsf{Rank}(T)$ to be $\mathsf{Dim}\{L_d\}_d$, which is the maximum number of independent linear polynomials among the factors of T. For a depth-5 circuit C, we shall use $\mathsf{Rank}(C)$ to denote $\max_{T \in \mathrm{Terms}(C)} \mathsf{Rank}(T)$.

For a parameter τ, we shall use $\mathrm{Terms}_{>\tau}(C)$ to refer to terms $T \in \mathrm{Terms}(C)$ with $\mathsf{Rank}(T) > \tau$.

Low rank gates are low-degree polynomials

The following Lemma, present implicitly in [6, 7], is a very useful transformation of gates of low-rank (and possibly large degree) when working over a finite field.

▶ **Lemma 8** ([6, 7]). *Let Q be a product of linear polynomials of rank at most τ. Then, there is a polynomial $\tilde{Q}$ of degree at most $(q-1)\cdot\tau$ such that $\tilde{Q}(\mathbf{a}) = Q(\mathbf{a})$ for all $\mathbf{a} \in \mathbb{F}_q^n$.*

Proof. Without loss of generality, we shall assume that the rank is equal to τ, as the degree upper bound will only be better for a smaller rank and let $L_1, \ldots, L_\tau$ be linearly independent. Let

$$Q = \prod_{i=[\tau]} L_i \cdot \prod_{j \notin [\tau]} L_j.$$

Here, each linear form in the second product term is in the linear span of the linear forms $\{L_i : i \in [\tau]\}$, and so can be expressed as their linear combination. Therefore, Q can be expressed as a polynomial in $\{L_i : i \in [\tau]\}$. Let $Q = f(L_1, L_2, \ldots, L_\tau)$. Since we are working over $\mathbb{F}_q$, it follows that for every choice of L_i and for every $\mathbf{a} \in \mathbb{F}_q^n$, we have $L_i^q(\mathbf{a}) = L_i(\mathbf{a})$. So, for every $\mathbf{a} \in \mathbb{F}_q^n$,

$$f(L_1, L_2, \ldots, L_\tau)(\mathbf{a}) = [f(L_1, L_2, \ldots, L_\tau) \mod \langle(\{L_i^q - L_i \ : \ i = 1, \ldots, \tau\})\rangle](\mathbf{a}).$$

The lemma immediately follows by setting

$$\tilde{Q} := f(L_1, L_2, \ldots, L_\tau) \mod \langle(\{L_i^q - L_i \ : \ i = 1, \ldots, \tau\})\rangle. \qquad \blacktriangleleft$$

High rank gates are almost always zero

Let us assume that $\mathrm{size}(C) \leq 2^{\sqrt{d}/100}$. We shall fix a threshold τ and call all terms T with $\mathsf{Rank}(T) > \tau$ as "high rank terms" and the rest as "low rank terms". Under a random evaluation in $\mathbb{F}_q^n$, every non-zero linear polynomial takes value zero with probability $1/q$. Thus, if we have a term that is a product of *many* independent linear polynomials, then with very high probability *many* of them will be set to zero, i.e. the term will vanish with high multiplicity at most points. This is formalized by the following definition and the lemma after it.

▶ **Definition 9** (Multiplicity at a point). For any polynomial P and a point $\mathbf{a} \in \mathbb{F}_q^n$, we shall say that $\mathbf{a}$ vanishes *with multiplicity* t on P if $Q(\mathbf{a}) = 0$ for all $Q \in \partial^{\leq t-1}(P)$. In other words, $\mathbf{a}$ is a root of P and all its derivatives up to order $t-1$.

We shall denote by $\mathsf{Mult}(P, \mathbf{a})$ the maximum t such that $\mathbf{a}$ vanishes on $\partial^{\leq t-1}(P)$.

It is easy to see that if P is a product of linear polynomials, then $\mathbf{a}$ vanishes with multiplicity t on P if $\mathbf{a}$ vanishes on at least t factors of P.

▶ **Observation 10.** *Let $T = \prod_{i=1}^d L_i$ be a term of rank at least r. Then, for every $\delta > 0$,*

$$\Pr_{\mathbf{a} \in \mathbb{F}_q^n}\left[\mathsf{Mult}(T, \mathbf{a}) \leq (1-\delta)\frac{r}{q}\right] \leq \exp\left(-\frac{\delta^2 r}{2q}\right).$$

Proof. Without loss of generality, let $L_1, \ldots, L_r$ be linearly independent. Then, the evaluations of $L_1, \ldots, L_r$ at a point $\mathbf{a} \in \mathbb{F}_q^n$ are also linearly independent and $\Pr_{\mathbf{a}}[L_i(\mathbf{a}) = 0] = (1/q)$ for $i = 1, \ldots, r$.

For $i = 1, \ldots, r$, let Y_i be the indicator random variable that is one if $L_i(\mathbf{a}) = 0$ and zero otherwise. Let $Y = \sum_{i \in [r]} Y_i$. Clearly, by linearity of expectations

$$\mathbb{E}[Y] = \sum_{i \in [r]} \mathbb{E}[Y_i] = \frac{r}{q}.$$

Since the events Y_i are linearly independent, by the Chernoff Bound, we know that for every $\delta > 0$

$$\Pr\left[Y \leq (1-\delta)\frac{r}{q}\right] \quad \leq \quad \exp\left(-\frac{\delta^2 r}{2q}\right). \qquad \blacktriangleleft$$

The following corollary is a simple union bound on all high-rank gates in a small circuit.

▶ **Corollary 11.** *Let C be a depth-5 circuit over $\mathbb{F}_q$ such that $\text{size}(C) \leq 2^{\sqrt{d}/100}$. Let $\tau = \frac{q\sqrt{d}}{6}$ so that*

$$\exp\left(\frac{\tau}{8 \cdot q}\right) \quad > \quad 2^{\sqrt{d}/50}.$$

Then,

$$\Pr_{\mathbf{a} \in \mathbb{F}_q^n}\left[\exists T \in \text{Terms}_{>\tau}(C) \ : \ \mathsf{Mult}(T, \mathbf{a}) \ \leq \ \frac{\tau}{2q}\right] \quad \leq \quad 2^{-(\sqrt{d}/100)}$$

We shall set our parameter τ as in the above corollary and our parameter $k = \tau/2q^3$.

5.1 Upper bound on complexity measure

For a circuit C of size at most $2^{\sqrt{d}/100}$, let $\mathcal{E}$ refer to the "bad set" of points $\mathbf{a}$ such that there is some $T \in \text{Terms}_{>\tau}(C)$ for which $\mathsf{Mult}(T, \mathbf{a}) \leq k = \tau/2q^3$. By the above corollary, we know that

$$|\mathcal{E}| = \delta \cdot q^n \quad \text{for some } \delta = \exp(-O(\sqrt{d})).$$

Let S be any subset of $\mathbb{F}_q^n \setminus \mathcal{E}$ that is contained in a "translate of a hypercube", that is there exists some $\mathbf{c} \in \mathbb{F}_q^n$ such that

$$S \subset (\mathbf{c} + \mathcal{H}) \setminus \mathcal{E}.$$

The following lemma allows us to "multilinearize" any polynomial as long as we are only interested in evaluations on a translate of a hypercube.

▶ **Lemma 12** (Multilinearization). *Fix a translate of a hypercube $\mathbf{c} + \mathcal{H}$. Then for every polynomial $Q \in \mathbb{F}_q[\mathbf{x}]$, there is a unique multilinear polynomial Q' such that $\mathsf{Deg}(Q') \leq \mathsf{Deg}(Q)$ and $Q'(\mathbf{a}) = Q(\mathbf{a})$ for every $\mathbf{a} \in \mathbf{c} + \mathcal{H}$.*

Proof. If $\mathbf{a} \in \mathbf{c} + \mathcal{H}$, then for each $i \in [n]$ we have a_i to be either c_i or $c_i + 1$. Thus, it suffices to replace each x_i^2 by a linear polynomial in x_i that maps c_i to c_i^2 and $c_i + 1$ to $(c_i + 1)^2$. This is achieved by $x_i^2 \mapsto c_i^2 + (x_i - c_i)(2c_i + 1)$. By repeated applications of this reduction, we obtain a multilinear polynomial Q' of degree at most $\mathsf{Deg}(Q)$ that agrees on all points on $\mathbf{c} + \mathcal{H}$.

Another way to state this is by looking at Q mod $\mathcal{I}_{\mathbf{c}}$ where $\mathcal{I}_{\mathbf{c}}$ is the ideal defined by

$$\mathcal{I}_{\mathbf{c}} \quad := \quad \left\langle \left(\left\{ x_i^2 - (c_i^2 + (x_i - c_i)(2c_i + 1)) \ : \ i = 1, \ldots, n \right\} \right\rangle \right).$$

It is easy to check that $\mathcal{I}_{\mathbf{c}}$ vanishes on $\mathbf{c} + \mathcal{H}$, and any Q can be reduced to a multilinear polynomial modulo $\mathcal{I}_{\mathbf{c}}$.

The uniqueness of Q' follows from the fact that no non-zero multilinear polynomial can vanish on all of $\mathbf{c} + \mathcal{H}$. ◀

We remark that *multilinearization* as defined above is similar to the notion of taking *multilinear projections* in [13]. However, the notion defined above is more amenable to work with when we are looking at evaluations of polynomials. It is not clear to us if the same can be done with the original notion of taking multilinear projections, as in [13].

The main lemma of this section would be the following bound on the complexity measure on a depth-5 circuit.

▶ **Lemma 13** (Upper bound on circuit)**.** *Let C be a depth-5 circuit, of formal degree at most $2d$ and $\text{size}(C) \leq 2^{\sqrt{d}/100}$, that computes an n-variate degree d polynomial. Let τ and k be chosen as above, and ℓ be a parameter satisfying $\ell + k\tau q < n/2$. If S is any subset of $\mathbb{F}_q^n \setminus \mathcal{E}$ that is contained in a translate of a hypercube, then*

$$\Gamma_{k,\ell,S}(C) \quad \leq \quad 2^{\sqrt{d}/100} \cdot \binom{\frac{4d}{\tau} + 1}{k} \cdot \binom{n}{\ell + k\tau q} \cdot \text{poly}(n).$$

Proof. Suppose $C = R_1 + \cdots + R_s$, where $s \leq 2^{\sqrt{d}/100}$ and each R_i is a product of depth-3 circuits with $\mathsf{Deg}(R_i) \leq 2d$. Since $\Gamma_{k,\ell,S}$ is clearly sub-additive (i.e. $\Gamma_{k,\ell,S}(f+g) \leq \Gamma_{k,\ell,S}(f) + \Gamma_{k,\ell,S}(g)$ for any f, g), it suffices to show that for each R_i we have

$$\Gamma_{k,\ell,S}(R_i) \quad \leq \quad \binom{\frac{4d}{\tau} + 1}{k} \cdot \binom{n}{\ell + k\tau q} \cdot \text{poly}(n).$$

For each such R_i, define the $R_i^{\leq \tau}$ as the polynomial obtained by "deleting" all terms $T \in \text{Terms}_{>\tau}(R_i)$. That is,

$$\text{if} \quad R_i \quad = \quad \prod_a \sum_b T_{ab} \quad \text{then} \quad R_i^{\leq \tau} \quad = \quad \prod_a \sum_{b : Rank(T_{ab}) \leq \tau} T_{ab}.$$

The lemma follows from the following two claims which we now prove.

▶ **Claim 14.** *For every $i \in [r]$*

$$\Gamma_{k,\ell,S}(R_i) \quad = \quad \Gamma_{k,\ell,S}(R_i^{\leq \tau}).$$

▶ **Claim 15.** *For every $i \in [r]$*

$$\Gamma_{k,\ell,S}(R_i^{\leq \tau}) \quad \leq \quad \binom{\frac{4d}{\tau} + 1}{k} \cdot \binom{n}{\ell + k\tau q} \cdot \text{poly}(n).$$ ◀

Proof of Claim 14. For brevity, we shall drop some indices and work with $R = Q_1 \cdots Q_m$. Let $T \in \text{Terms}_{>\tau}(C)$. We shall show if $R' = (Q_1 - T)Q_2 \cdots Q_m$, then for any k-th order partial derivative $\partial_{\mathbf{x}^\alpha}$,

$$\mathsf{Eval}_S(\partial_{\mathbf{x}^\alpha}(R)) \quad = \mathsf{Eval}_S(\partial_{\mathbf{x}^\alpha}(R')).$$

Indeed, consider the difference $R - R' = T \cdot Q_2 \cdots Q_m$. By the chain rule, every term in $\partial_{\mathbf{x}^\alpha}(R - R')$ is divisible by some k'-th order partial derivative of T with $k' \leq k$. By the choice of S, we know that every $\mathbf{a} \in S$ satisfies $\mathsf{Mult}(T, \mathbf{a}) > k$, and hence $\mathbf{a}$ vanishes on $\partial^{\leq k}(T)$ for any $T \in \mathrm{Terms}_{>\tau}(C)$. Thus, it follows that $\mathsf{Eval}_S(\partial_{\mathbf{x}^\alpha}(R - R'))$ is just the zero vector.

Repeating this argument, we can prune away all terms in $\mathrm{Terms}_{>\tau}(C)$ to get that $\mathsf{Eval}_S(\partial_\alpha(R)) = \mathsf{Eval}_S(\partial_{\mathbf{x}^\alpha}(R^{\leq\tau}))$ where $\mathsf{Deg}(\mathbf{x}^\alpha) = k$. Thus, $\Gamma_{k,\ell,S}(R) = \Gamma_{k,\ell,S}(R^{\leq\tau})$. ◀

Proof of Claim 15. Let $R^{\leq\tau} = Q_1 \cdots Q_d$, with each Q_i being a $\Sigma\Pi\Sigma$ circuit. Some of these Q_is could have degree more than τ although their rank is bounded by τ. Without loss of generality, let $Q_1, \cdots, Q_m$ be all the Q_is with $\mathsf{Deg}(Q_i) > \tau$, and $Q_{m+1}, \ldots, Q_d$ have $\mathsf{Deg}(Q_i) \leq \tau$.

We shall modify the "low-degree" Q_is by multiplying together any two of them of degree less than $\tau/2$. This ensures that at most one of the Q_is may have degree less than $\tau/2$ and for $i > m$, all the Q_is have degree at most τ. The sizes of some of the low-degree Q_is do increase in the process but this would not be critical as the degree of any such term is still bounded by τ. The main point is that now we have an expression of the form

$$R^{\leq\tau} = Q_1 \cdots Q_m \cdot Q'_1 \cdots Q'_r$$

where each Q_i is a $\Sigma\Pi\Sigma$ circuit of rank at most $\tau - 1$ and $\mathsf{Deg}(Q_i) \geq \tau$, and all but one of the Q'_i satisfies $\tau \geq \mathsf{Deg}(Q'_i) \geq \tau/2$. As $\mathsf{Deg}(R^{\leq\tau}) \leq 2d$, it follows that $m + r \leq \frac{4d}{\tau} + 1$.

As a notational convenience, for any set H we let $Q_H := \prod_{i\in H} Q_i$ and we use $R^{\leq\tau}$ and R interchangeably in the calculations that follow. Let us look at any partial derivative $\partial_{\mathbf{x}^\alpha}$ of order k applied on R. By the chain-rule, any such partial derivative can be written as a natural linear combination of terms. For any two monomials $\mathbf{x}^\alpha = \prod_{i=1}^n x_i^{\alpha_1}$ and $\mathbf{x}^\beta = \prod_{i=1}^n x_i^{\beta_i}$, we say that $\mathbf{x}^\alpha \succeq \mathbf{x}^\beta$, if there exists a monomial $\mathbf{x}^\gamma = \prod_{i=1}^n x_i^{\gamma_i}$ such that $\mathbf{x}^\alpha = \mathbf{x}^\beta \cdot \mathbf{x}^\gamma$.

$$\begin{aligned}
\partial_{\mathbf{x}^\alpha}(R) &\in \mathsf{Span}\left\{ \partial_{\mathbf{x}^\beta}(Q_A) \cdot \partial_{\mathbf{x}^\gamma}(Q'_B) \cdot Q_{\overline{A}} \cdot Q'_{\overline{B}} \;:\; \begin{array}{c} \mathbf{x}^\alpha = \mathbf{x}^\beta \cdot \mathbf{x}^\gamma\,,\ A \subseteq [m]\,, \\ B \subseteq [r]\,,\ |A| + |B| = k \end{array} \right\} \\
&\in \mathsf{Span}\left\{ \partial_{\mathbf{x}^\beta}(Q_A) \cdot \mathbf{x}^{\leq k\tau} \cdot Q_{\overline{A}} \cdot Q'_{\overline{B}} \;:\; \begin{array}{c} \mathbf{x}^\alpha \succeq \mathbf{x}^\beta\,,\ A \subseteq [m]\,, \\ B \subseteq [r]\,,\ |A| + |B| = k \end{array} \right\}. \\
\mathbf{x}^{=\ell}\partial_{\mathbf{x}^\alpha}(R) &\subseteq \mathsf{Span}\left\{ \partial_{\mathbf{x}^\beta}(Q_A) \cdot \mathbf{x}^{\leq \ell + k\tau} \cdot Q_{\overline{A}} \cdot Q'_{\overline{B}} \;:\; \begin{array}{c} \mathbf{x}^\alpha \succeq \mathbf{x}^\beta\,,\ A \subseteq [m]\,, \\ B \subseteq [r]\,,\ |A| + |B| = k \end{array} \right\}. \\
\mathsf{Eval}_S(\mathbf{x}^{=\ell}\partial_{\mathbf{x}^\alpha}(R)) &\subseteq \mathsf{Span}\left\{ \mathsf{Eval}_S\left(\partial_{\mathbf{x}^\beta}(Q_A) \cdot \mathbf{x}^{\leq \ell + k\tau} \cdot Q_{\overline{A}} \cdot Q'_{\overline{B}}\right) \;:\; \begin{array}{c} \mathbf{x}^\alpha \succeq \mathbf{x}^\beta\,,\ A \subseteq [m]\,, \\ B \subseteq [r]\,,\ |A| + |B| = k \end{array} \right\}.
\end{aligned}$$

If we focus on the term $\partial_{\mathbf{x}^\beta}(Q_A)$, since Q_A is a product of $\Sigma\Pi\Sigma$ circuits of rank at most τ, we have that $\partial_{\mathbf{x}^\beta}(Q_A)$ is a linear combination of terms $T_1 \cdots T_{|A|}$ where each T_i is a product of linear polynomials and has rank at most τ. Using Lemma 8 on each of these T_is,

$$\mathsf{Eval}_S(\partial_{\mathbf{x}^\beta}(Q_A)) \in \mathsf{Span}\left\{ \mathsf{Eval}_S(\mathbf{x}^{\leq(q-1)\tau|A|}) \right\}.$$

Therefore,

$$\begin{aligned}
\mathsf{Eval}_S(\mathbf{x}^{=\ell}\partial_{\mathbf{x}^\alpha}(R)) &\subseteq \mathsf{Span}\left\{ \mathsf{Eval}_S\left(\partial_{\mathbf{x}^\beta}(Q_A) \cdot \mathbf{x}^{\leq \ell + k\tau} \cdot Q_{\overline{A}} \cdot Q'_{\overline{B}}\right) \;:\; \begin{array}{c} \mathbf{x}^\alpha \succeq \mathbf{x}^\beta\,,\ A \subseteq [m]\,, \\ B \subseteq [r]\,,\ |A| + |B| = k \end{array} \right\} \\
&\subseteq \mathsf{Span}\left\{ \mathsf{Eval}_S\left(\mathbf{x}^{\leq \ell + k\tau + (q-1)k\tau} \cdot Q_{\overline{A}} \cdot Q'_{\overline{B}}\right) \;:\; \begin{array}{c} A \subseteq [m]\,, \\ B \subseteq [r]\,,\ |A| + |B| = k \end{array} \right\}.
\end{aligned}$$

Finally, Lemma 12 shows for every polynomial f, there is a multilinear polynomial of degree at most $\mathsf{Deg}(f)$ that agrees with f on all evaluations on a translate of a hypercube. Therefore,

in the above span, we may assume that we are only shifting by multilinear monomials of degree $\ell + qk\tau$ as this doesn't change the evaluations $S \subseteq \mathbf{c} + \{0,1\}^n$. Hence,

$$\mathsf{Eval}_S(\mathbf{x}^{=\ell}\partial_{\mathbf{x}^\alpha}(R)) \subseteq \mathsf{Span}\left\{ \mathsf{Eval}_S\left(\mathbf{x}_{\text{mult}}^{\leq \ell+qk\tau} \cdot Q_{\overline{A}} \cdot Q'_{\overline{B}}\right) \;:\; \begin{array}{c} A \subseteq [m]\,, \\ B \subseteq [r]\,,\ |A|+|B| = k \end{array} \right\}.$$

Therefore, using the fact that $m + r \leq (4d/\tau) + 1$, we get the bound

$$\Gamma_{k,\ell,S}(R) := \mathsf{Dim}\left\{\mathsf{Eval}_S(\mathbf{x}^{=\ell}\partial^{=k}(R))\right\} \quad \leq \quad \binom{\frac{4d}{\tau}+1}{k} \cdot \binom{n}{\ell+qk\tau} \cdot n,$$

where the first term corresponds to the number of choices for the subsets A and B, and the last two terms correspond to the number of multilinear monomials of degree at most $\ell + qk\tau$. Recall that by the hypothesis of the lemma, $\ell + qk\tau < n/2$, hence the number of multilinear monomials of degree at most $\ell + qk\tau$ is at most $n \cdot \binom{n}{\ell+qk\tau}$. ◀

▶ Remark 16. Observe that, even if the circuit C is of the form

$$C \quad = \quad \sum_a \prod_{b\in[m]} \sum_c \prod_d L_{abcd}$$

such that $\text{Size}(C) \leq 2^{\sqrt{d}/100}$ and $m = O(\frac{d}{\tau})$, then the upper bound in Lemma 13 continues to hold.[7] In particular, the formal degree of C could be much larger than d but if the product fan-in at level two of C is small, then

$$\Gamma_{k,\ell,S}(C) \quad \leq \quad 2^{\sqrt{d}/100} \cdot \binom{O(\frac{d}{\tau})}{k} \cdot \binom{n}{\ell+k\tau q} \cdot \text{poly}(n).$$

We later use this observation to complete the proof of Theorem 2 in Section 7.

6 Lower bound for the complexity measure for an explicit polynomial

Let $\mathcal{E}$ be an arbitrary subset of $\mathbb{F}_q^n$ of size at most $\delta \cdot q^n$. We will be choosing a specific set S that shall be a subset of a translate of the hypercube and disjoint from $\mathcal{E}$. We will fix the precise definition of S shortly once we motivate the requirements.

The polynomial for which was shall prove our lower bound would be from the Nisan-Wigderson family. We would have to set our parameters carefully but for now we shall just be intentionally vague and refer to the polynomial as just NW and fix parameters at a later point.

Associated with our measure $\Gamma_{k,\ell,S}(\mathsf{NW})$ is a natural matrix that we shall call $\Lambda(\mathsf{NW})$:

> The rows of $\Lambda(\mathsf{NW})$ are indexed by a derivative $\partial_{\mathbf{x}^\alpha} \in \partial^{=k}$ of order k, and a monomial $\mathbf{x}^\beta$ of degree equal to ℓ. The columns are indexed by all points $\mathbf{a} \in S$. The entry in $(\mathbf{x}^\beta \cdot \partial_{\mathbf{x}^\alpha}, \mathbf{a})$ is the evaluation of $\mathbf{x}^\beta \cdot \partial_{\mathbf{x}^\alpha}(\mathsf{NW})$ at the point $\mathbf{a}$.

In other words, the matrix is just the vectors $\mathsf{Eval}_S(\mathbf{x}^\beta \cdot \partial_{\mathbf{x}^\alpha}(\mathsf{NW}))$ listed as different rows for each choice of $\mathbf{x}^\alpha$ and $\mathbf{x}^\beta$. Therefore,

$$\mathsf{Rank}\left(\Lambda(\mathsf{NW})\right) = \Gamma_{k,\ell,S}(\mathsf{NW}). \tag{6.1}$$

[7] Essentially, in the proof of Claim 15, we already have an expression of the form $R^{\leq\tau} = Q_1 \cdots Q_m$ with $m = O\left(\frac{d}{\tau}\right)$ and the rest of the proof proceeds as expected.

Recall from Lemma 12 (multilinearization), as long as we only care about evaluations on a translate of a hypercube, we can assume each row is the evaluations of the multilinearization of $\mathbf{x}^\alpha \cdot \partial_{\mathbf{x}^\beta}(\mathsf{NW})$. This does not change the evaluation on any point $\mathbf{a} \in S \subseteq \mathbf{c} + \mathcal{H}$.

Now any such matrix of evaluations can be naturally factorized as a coefficient matrix and an evaluation matrix.

$C_{k,\ell}(\mathsf{NW})$: Each row is indexed by a derivative $\partial_{\mathbf{x}^\alpha}$ of order k, and a monomial $\mathbf{x}^\beta$ of degree ℓ, and each column is indexed by a multilinear monomial m of degree *at most* $\ell + d - k$ over n variables, and the $(\mathbf{x}^\beta \cdot \partial_{\mathbf{x}^\alpha}, m)$ entry is the coefficient of monomial m in the multilinearization of $\mathbf{x}^\beta \cdot \partial_{\mathbf{x}^\alpha}(\mathsf{NW})$ with respect to $\mathbf{c} + \mathcal{H}$ (Lemma 12).

$V_t(S)$: Rows are indexed by multilinear monomials of degree at most $t = \ell + d - k$ over n variables, columns are indexed by $\mathbf{a} \in S$ and $(m, \mathbf{a})$ entry is the evaluation monomial m at $\mathbf{a}$.

Clearly, $\Lambda(\mathsf{NW}) = C_{k,\ell}(\mathsf{NW}) \cdot V_t(S)$. Thus if we can get a good lower bound on the ranks of the matrices $C_{k,\ell}(\mathsf{NW})$ and $V_t(S)$ for a suitable set S, we would then be able to lower bound the rank of $\Lambda(\mathsf{NW})$. This is formalized by the following simple linear algebraic fact often referred to as the Sylvester's rank inequality.

▶ **Lemma 17** (Rank of products of matrices [27]). *If A, B and C are matrices such that $A = B \cdot C$, then* $\mathsf{Rank}(A) \geq \mathsf{Rank}(B) + \mathsf{Rank}(C) - (\#\ \textit{rows of}\ C)$.

6.1 Rank of $C_{k,\ell}(\mathsf{NW})$

Let us focus on the matrix $C_{k,\ell}(\mathsf{NW})$ and restrict ourselves a sub-matrix $C'_{k,\ell}(\mathsf{NW})$ to only those columns whose degree is *exactly* $t = \ell + d - k$, and rows indexed by $(\mathbf{x}^\beta \cdot \partial_{\mathbf{x}^\alpha})$ with $\mathbf{x}^\beta$ being a multilinear monomial of degree exactly ℓ.

Since our polynomial NW is multilinear, if we were to read off any row of $C'_{k,\ell}(\mathsf{NW})$, this is just the list of coefficients of all multilinear monomials of $(\mathbf{x}^\beta \cdot \partial_{\mathbf{x}^\alpha}(\mathsf{NW}))$. This is because the multilinearization operation in Lemma 12 maps any non-multilinear monomial to a multilinear polynomial of strictly smaller degree and hence these monomials are not included in the columns of $C'_{k,\ell}$.

The key point here is that the matrix $C'_{k,\ell}(\mathsf{NW})$ is just the matrix of *projected shifted partial derivatives* of NW. The results of Kayal et. al [13] and Kumar and Saraf [19] give a lower bound on the rank of this matrix for a suitable choice of the polynomial, but the lower bound of Kumar and Saraf [19] is more relevant as it is true over any field (unlike [13] that works only over characteristic zero fields).

Using a tight analysis of the argument in [19], that we present in Section A we obtain the following lemma for the Nisan-Wigderson polynomial for very carefully chosen parameters.

▶ **Lemma 18** (Tight analysis of [19]). *For every d and $k = O(\sqrt{d})$ there exists parameters m, e, ϵ such that $m = \Theta(d^2)$, $n = md$ and $\epsilon = \Theta\left(\frac{\log d}{\sqrt{d}}\right)$ with*

$$\begin{aligned} m^k &\geq (1+\epsilon)^{2(d-k)} \\ m^{e-k} &= \left(\frac{2}{1+\epsilon}\right)^{d-k} \cdot \mathrm{poly}(m). \end{aligned}$$

For any $\{d, m, e, k, \epsilon\}$ satisfying the above constraints and for $\ell = \frac{n}{2}(1-\epsilon)$, over any field $\mathbb{F}$, we have

$$\mathsf{Rank}(C_{k,\ell}(\mathsf{NW}_{d,m,e})) \quad \geq \quad \mathsf{Rank}(C'_{k,\ell}(\mathsf{NW}_{d,m,e})) \quad \geq \quad \binom{n}{\ell+d-k} \cdot \exp(-O(\log^2 d)).$$

6.2 Rank of $V_t(S)$

Let $\mathcal{H}_{\leq t}$ refer to elements of $\{0,1\}^n$ of Hamming weight at most t. Our first step would be to choose our set S carefully so that we can maximize the rank of $V_t(S)$.

▶ **Observation 19.** *Let $\mathcal{E}$ be a subset of $\mathbb{F}_q^n$ of size at most $\delta \cdot q^n$. Then for any $0 \leq t \leq n$, there is a vector $\mathbf{c} \in \mathbb{F}_q^n$ such that*

$$|(\mathbf{c} + \mathcal{H}_{\leq t}) \cap \mathcal{E}| \quad \leq \quad \delta \cdot |\mathcal{H}_{\leq t}| \,.$$

Proof. Let $\mathbb{1}_{\mathcal{E}}(\mathbf{a})$ be the indicator function that is 1 if $\mathbf{a} \in \mathcal{E}$, and 0 otherwise. Then,

$$\delta \quad \geq \quad \mathbb{E}_{\mathbf{a} \in \mathbb{F}_q^n} \left[\mathbb{1}_{\mathcal{E}}(\mathbf{a})\right] \quad = \quad \mathbb{E}_{\mathbf{c} \in \mathbb{F}_q^n} \left[\mathbb{E}_{\mathbf{y} \in \mathcal{H}_{\leq t}} \left[\mathbb{1}_{\mathcal{E}}(\mathbf{c}+\mathbf{y})\right]\right] .$$

Thus, there exists some $\mathbf{c} \in \mathbb{F}_q^n$ that still maintains the inequality. ◀

Our set would be $S = (\mathbf{c}+\mathcal{H}_{\leq t}) \setminus \mathcal{E}$, which has the property that $|S \cap (\mathbf{c} + \mathcal{H}_{\leq t})| \geq (1-\delta) \cdot |\mathcal{H}_{\leq t}|$ by the above observation, and $S \cap \mathcal{E} = \emptyset$.

Let $V_t(S - \mathbf{c})$ be the matrix whose rows are indexed by the polynomials $m(\mathbf{x} - \mathbf{c})$, where m is a multilinear monomials in variables $\mathbf{x}$ of degree at most t. The columns of $V_t(S - \mathbf{c})$ are indexed by S and the entries correspond to the evaluation of the polynomial indexing the row at the point in S given by the column. We have the following observation.

▶ **Observation 20.** $\mathsf{Rank}(V_t(S)) = \mathsf{Rank}(V_t(S - \mathbf{c}))$.

Proof. For any multilinear monomial m of degree at most t, the polynomial $m(\mathbf{x} - \mathbf{c})$ is multilinear and has degree at most t. Thus clearly, the row-space of $V_t(S - \mathbf{c})$ is contained in the row-space of $V_t(S)$. The converse also holds trivially as the translation is invertible. ◀

We now prove our next lemma which shows a lower bound on the rank of $V_t(S - \mathbf{c})$.

▶ **Lemma 21.** *For any set $S \subseteq \{0,1\}^n \subset \mathbb{F}_q^n$ and any $0 \leq t \leq n$,*

$$\mathsf{Rank}(V_t(S - \mathbf{c})) \quad = \quad |S| \,.$$

Proof. Since $S \subseteq \mathbf{c} + \mathcal{H}_{\leq t}$, the set $S' := S - \mathbf{c} \subset \mathcal{H}_{\leq t}$. Thus the matrix $V_t(S - \mathbf{c})$ is simply the matrix $V_t(S')$. For any $\mathbf{a} \in \{0,1\}^n$, let $m_{\mathbf{a}}$ refer to the 'characteristic' monomial $\prod_{i:a_i=1} x_i$, and let $m_{\mathbf{0}} = 1$.

Consider the sub-matrix of $V_t(S')$ by restricting to rows indexed by $\{m_{\mathbf{a}} \,:\, \mathbf{a} \in S'\}$. By rearranging the rows and columns in the increasing order of the weight of $\mathbf{a}$, it is evident that the sub-matrix is upper-triangular with ones on the diagonal. It therefore follows that the rank of $V_t(S')$ (which is just $V_t(S - \mathbf{c})$) is at least $|S'| = |S|$. ◀

Combining Observation 20 and Lemma 21, we have our required bound on the rank of $V_t(S)$.

▶ **Lemma 22.** *Let $\mathcal{E}$ be an arbitrary subset of $\mathbb{F}_q^n$ of size at most $\delta \cdot q^n$. Then, there exists a set $S \subset \mathbb{F}_q^n \setminus \mathcal{E}$ such that $S \subseteq \mathbf{c} + \mathcal{H}$ for some $\mathbf{c} \in \mathbb{F}_q^n$ for which*

$$\mathsf{Rank}(V_t(S)) \quad \geq \quad (1-\delta) \cdot |\mathcal{H}_{\leq t}| \quad = \quad (1-\delta) \cdot (\#\text{ rows of } V_t(S)).$$

Putting them together

▶ **Lemma 23** (Rank bound for $\Lambda(\mathsf{NW}_{d,m,e})$)**.** *Let $\mathcal{E}$ be an arbitrary subset of $\mathbb{F}_q^n$ of size at most $\delta \cdot q^n$, with $\delta = \exp(-\omega(\log^2 d))$. Then, there exists a set $S \subset \mathbb{F}_q^n \setminus \mathcal{E}$ such that $S \subseteq \mathbf{c} + \mathcal{H}$ for some $\mathbf{c} \in \mathbb{F}_q^n$ for which*

$$\mathsf{Rank}(\Lambda(\mathsf{NW}_{d,m,e})) \quad \geq \quad \binom{n}{\ell + d - k} \cdot \exp(-O(\log^2 d)),$$

where the parameters d, m, e, k, ℓ are chosen as in Lemma 18.

Proof. Consider the set S chosen in Lemma 22 (for $t = \ell + d - k$). By Lemma 22,

$$\mathsf{Rank}(V_t(S)) - (\#\text{ rows of } V_t(S)) \quad \geq \quad (-\delta)\,|\mathcal{H}_{\leq t}| \quad \geq \quad (-\delta) \cdot n \cdot \binom{n}{\ell + d - k}.$$

Lemma 18 shows that rank of $C_{k,\ell}(\mathsf{NW}_{d,m,e})$ can be lower bounded by

$$\mathsf{Rank}(C_{k,\ell}(\mathsf{NW}_{d,m,e})) \quad \geq \quad \binom{n}{\ell + d - k} \cdot \exp(-O(\log^2 d)).$$

Thus, since $\Lambda(\mathsf{NW}_{d,m,e}) = C_{k,\ell}(\mathsf{NW}_{d,m,e}) \cdot V_t(S)$ with $t = \ell + d - k$, Lemma 17 implies that

$$\begin{aligned}
\mathsf{Rank}(\Lambda(\mathsf{NW}_{d,m,e})) &\geq \mathsf{Rank}(C_{k,\ell}(\mathsf{NW}_{d,m,e})) + \mathsf{Rank}(V_t(S)) - (\#\text{ rows of } V_t(S)) \\
&\geq \binom{n}{\ell + d - k} \cdot \exp(-O(\log^2 d)) - \delta \cdot n \cdot \binom{n}{\ell + d - k} \\
&\geq \binom{n}{\ell + d - k} \cdot \exp(-O(\log^2 d)) \qquad \text{as } \delta = \exp(-\omega(\log^2 d)). \quad \blacktriangleleft
\end{aligned}$$

Combining this with Equation 6.1, we get the following lemma.

▶ **Lemma 24** (Measure of $\mathsf{NW}_{d,m,e}$)**.** *Let $\mathcal{E}$ be an arbitrary subset of $\mathbb{F}_q^n$ of size at most $\delta \cdot q^n$, with $\delta = \exp(-\omega(\log^2 d))$. Then, there exists a set $S \subset \mathbb{F}_q^n \setminus \mathcal{E}$ such that $S \subseteq \mathbf{c} + \mathcal{H}$ for some $\mathbf{c} \in \mathbb{F}_q^n$ for which*

$$\Gamma_{k,\ell,S}(\mathsf{NW}_{d,m,e}) \quad \geq \quad \binom{n}{\ell + d - k} \cdot \exp(-O(\log^2 d)).$$

7 Wrapping up the proof

▶ **Theorem 25.** *Let $\mathbb{F}_q$ be the finite field of cardinality q. Let C be a depth-5 circuit of formal degree at most $2d$ which computes the polynomial $\mathsf{NW}_{d,m,e}$ with parameters as in Lemma 18. Then*

$$\mathit{Size}(C) > 2^{\sqrt{d}/100}.$$

Further, the same lower bound holds even if C was a circuit of the form

$$C \quad = \quad \sum_i \prod_{j \in [m]} \sum_k \prod_\ell L_{ijk\ell},$$

with $m = O(\sqrt{d})$.

Proof. We shall prove the above theorem for homogeneous depth-5 circuits. The lower bound for such non-homogeneous circuits would also follow directly since such circuits also have the same upper-bound on the complexity measure (Remark 16).

Assume on the contrary that there is a circuit C computing $\mathsf{NW}_{d,m,e}$ with $\text{Size}(C) \leq 2^{\sqrt{d}/100}$. Let τ be as defined in Corollary 11 and let $k = \tau/2q^3$. Let $\mathcal{E} = \mathcal{E}(\mathcal{C})$ be the set as defined in Section 5.1. We know that

$$|\mathcal{E}| \leq \delta \cdot q^n,$$

for some $\delta = \exp(-O(\sqrt{d}))$. Let $\ell = \frac{n}{2}(1-\epsilon)$ where $\epsilon = \frac{\log d}{c\sqrt{d}}$ is chosen as in Lemma 18. Since $n = d^3$, clearly we satisfy $\ell + k\tau q < n/2$. Let $S \subset \mathbb{F}_q^n \setminus \mathcal{E}$ be the set guaranteed by Lemma 24. From Lemma 24, we know that

$$\Gamma_{k,\ell,S}(\mathsf{NW}) \geq \binom{n}{\ell + d - k} \cdot \exp(-O(\log^2 d)).$$

This may be simplified using Lemma 32 to

$$\Gamma_{k,\ell,S}(\mathsf{NW}) \geq \binom{n}{\ell} \cdot (1+\epsilon)^{2d-2k} \cdot \exp(-O(d\epsilon^2)) \cdot \exp(-O(\log^2 d)).$$

Also, from Lemma 13, we know that

$$\Gamma_{k,\ell,S}(C) \leq 2^{\sqrt{d}/100} \cdot \binom{\frac{4d}{\tau} + 1}{k} \cdot \binom{n}{\ell + qk\tau} \cdot \text{poly}(n).$$

Notice that from our choice of k and τ, $qk\tau = O(d) = O(\sqrt{n})$. Again, using Lemma 32, we get

$$\Gamma_{k,\ell,S}(C) \leq 2^{\sqrt{d}/100} \cdot \binom{\frac{4d}{\tau} + 1}{k} \cdot \binom{n}{\ell} \cdot (1+\epsilon)^{2qk\tau} \cdot \exp(O(-qk\tau \cdot \epsilon^2)) \cdot \text{poly}(n).$$

Since C computes $\mathsf{NW}_{d,m,e}$, so it must be the case that

$$2^{\sqrt{d}/100} \cdot \text{poly}(n) \quad \geq \quad (1+\epsilon)^{(d-k)+(d-k-2qk\tau)} \cdot \exp(-O_q(\log^2 d)).$$

Since $k = \tau/2q^3$, so $2qk\tau = \tau^2/q^2$. From our choice of τ in Corollary 11, $\tau = \frac{q\sqrt{d}}{6}$. So

$$2qk\tau = \tau^2/q^2 = d/36.$$

Therefore,

$$2^{\sqrt{d}/100} \cdot \text{poly}(n) \geq (1+\epsilon)^{(d-k)} \cdot \exp(-O_q(\log^2 d)).$$

But this is a contradiction since $(1+\epsilon)^{(d-k)} = \exp(\Omega(\sqrt{d}\log d))$ by our choice of parameters. Therefore, the size of C is at least $2^{\sqrt{d}/100}$. ◀

In fact, the above proof gives more. It shows that if we have a depth-5 circuit computing $\mathsf{NW}_{d,m,e}$ over $\mathbb{F}_q$, then either the number of high-rank terms is at least $2^{\sqrt{d}/50}$ or the top fan-in is $\exp(\Omega(\sqrt{d}\log d))$.

7.1 Getting the right order of quantifiers

In our proof so far, we first fix the field $\mathbb{F}_q$ that we are working over and the parameters of our polynomial are then chosen based on q. Thus, as q varies, the polynomial for which we show the lower bound also seems to vary. The ideal scenario would be to construct a fixed polynomial family so that for every q we get a lower bound of $\exp(\Omega_q(\sqrt{d}))$. We do that now, and this would complete the proof of Theorem 1.

We shall be dealing with a lot of parameters and constraints. The following is essentially the "zone" in which we can prove strong lower bounds (Lemma 18).

▶ **Definition 26** (Goldilocks Zone)**.** We shall say that parameters m, d, e, k, ϵ with $k = \Theta(\sqrt{d})$ and $\epsilon = \Theta\left(\frac{\log d}{\sqrt{d}}\right)$ lie in the *Goldilocks Zone* if they satisfy

$$\begin{aligned} m^k &\geq (1+\epsilon)^{2(d-k)} \\ m^{e-k} &= \left(\frac{2}{1+\epsilon}\right)^{d-k} \cdot \mathrm{poly}(m). \end{aligned}$$

Recall that for Lemma 18, and consequently Theorem 25, the parameters m, d, e, k must lie in the Goldilocks zone. The crucial point is that this is a field dependent condition since k (and hence everything else) explicitly depends on q. In the next lemma, we show that we can start with a fixed polynomial such that for every finite field $\mathbb{F}_q$ of fixed size, there exists a $0,1$ projection which lies in the Goldilocks zone.

▶ **Lemma 27.** *Consider the* $\mathsf{NW}_{d,m,e}$ *polynomial with* $m = \Theta(d^2)$ *and* $e = \left\lceil \frac{d}{\log m} \right\rceil$ *so that*

$$m^e = 2^d \cdot \mathrm{poly}(m).$$

Suppose $k = \Theta(\sqrt{d})$ *and* $\epsilon = \Theta\left(\frac{\log d}{\sqrt{d}}\right)$ *satisfy the constraint* $m^k > (1+\epsilon)^{2(d-k)}$*. Then, there exists a* $d' \in [d - O(\sqrt{d}\log d), d]$ *such that* $\mathsf{NW}_{d',m,e}$ *is a* $0/1$ *projection of* $\mathsf{NW}_{d,m,e}$ *and the parameters* $\{d', m, e, k, \epsilon\}$ *fall in the Goldilocks Zone.*

Proof. Since $m^e = 2^d \cdot \mathrm{poly}(m)$, $m^k > (1+\epsilon)^{2(d-k)}$ and $(1+\epsilon)^d = \exp(\Theta(\sqrt{d}\log d))$, we have

$$m^{e-k} = \left(\frac{2}{1+\epsilon}\right)^{d-k} \cdot \exp(-\Theta(\sqrt{d}\log d)).$$

The goal now is to find a $d' < d$ such that

$$m^{e-k} = \left(\frac{2}{1+\epsilon}\right)^{d'-k} \cdot O(1).$$

Indeed, since the LHS and RHS differ by just a factor of $\exp(\Theta(\sqrt{d}\log d))$, and decreasing d by 1 reduces the RHS by a constant factor, there exists some $d' \in [d - O(\sqrt{d}\log d), d]$ such that

$$m^{e-k} = \left(\frac{2}{1+\epsilon}\right)^{d'-k} \cdot O(1).$$

Further, since $m^k > (1+\epsilon)^{2(d-k)}$, it follows that $m^k > (1+\epsilon)^{2(d'-k)}$ as $d' < d$. Hence the parameters $\{d', m, e, k, \epsilon\}$ indeed fall in the Goldilocks Zone (Definition 26).

It suffices to show that $\mathsf{NW}_{d',m,e}$ is a projection of $\mathsf{NW}_{d,m,e}$. This is readily seen as setting the variables $x_{ij} = 1$ for all $i \in [d - d']$ and $j \in [m]$ yields $\mathsf{NW}_{d',m,e}$ up to relabeling variables. ◀

With this, we can finally prove our main theorems.

▶ **Theorem 1** (restated). *Consider the polynomial* $\mathsf{NW}_{d,m,e}$ *with parameters chosen such that* $m = \Theta(d^2)$ *and* $m^e = 2^d \cdot \mathrm{poly}(m)$*. Then, for any fixed finite field* $\mathbb{F}_q$*, any homogeneous depth-5 circuit over* $\mathbb{F}_q$ *computing* $\mathsf{NW}_{d,m,e}$ *must have size at least* $2^{\sqrt{d}/200}$*.*

Proof. Fix a field $\mathbb{F}_q$ and let $k = \sqrt{d}/12q^2$.

Suppose on the contrary that there is indeed a homogeneous depth-5 circuit C computing $\mathsf{NW}_{d,m,e}$. Then, by Lemma 27, this also implies there is a projection C' that computes the $\mathsf{NW}_{d',m,e}$ such that there is an $d - O(\sqrt{d}\log d) \leq d' \leq d$ and there is an $\epsilon = \Theta\left(\frac{\log d}{\sqrt{d}}\right)$ for which $\{d', m, e, k, \epsilon\}$ fall in the Goldilocks Zone (Definition 26). Now C' is a circuit of formal degree $d \leq d' + O(\sqrt{d}\log d) \leq 2d'$ that computes the polynomial $\mathsf{NW}_{d',m,e}$. By Theorem 25, this implies that

$$\mathrm{size}(C) \geq \mathrm{size}(C') > 2^{\sqrt{d'}/100} > 2^{\sqrt{d}/200}.$$ ◀

The proof of this theorem also follows along the same lines.

▶ **Theorem 2** (restated). *Consider the polynomial* $\mathsf{NW}_{d,m,e}$ *with parameters chosen such that* $m = \Theta(d^2)$ *and* $m^e = 2^d \cdot \mathrm{poly}(m)$*. Then, for any fixed finite field* $\mathbb{F}_q$*, any depth-5 circuit over* $\mathbb{F}_q$ *of the form*

$$C \quad = \quad \sum_i \prod_{j\in[m]} \sum_k \prod_\ell L_{ijk\ell}$$

where each $L_{ijk\ell}$ *is a linear polynomial and* $m = O(\sqrt{d})$ *that computes* $\mathsf{NW}_{d,m,e}$ *must have size at least* $2^{\sqrt{d}/200}$*.*

Acknowledgments. We are very grateful to Mike Saks and Shubhangi Saraf for many discussions and much encouragement. The chat with Mike about powering circuits over finite fields was specially insightful. Part of this work was done while the first author was an intern at MSR New England. We are thankful to Madhu Sudan and the other members of the lab, for stimulating discussions and generous hospitality. Many thanks to Madhu for patiently sitting through a presentation of the proof.

We would also like to thank Eric Allender for answering our questions about connections between boolean circuits and arithmetic circuits over finite fields and pointing us to reference [1].

This work was partly motivated by discussions over the questions of the projected shifted partials complexity of homogeneous depth-5 circuits while the authors were at MSR Bangalore in Summer'14. We are grateful to Neeraj Kayal for hosting us and for many insightful conversations.

References

1 Manindra Agrawal, Eric Allender, and Samir Datta. On TC^0, AC^0, and Arithmetic Circuits. *Journal of Computer and System Sciences*, 60(2):395–421, 2000. doi:10.1006/jcss.1999.1675.

2 Manindra Agrawal and V. Vinay. Arithmetic circuits: A chasm at depth four. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2008)*, pages 67–75, 2008. doi:10.1109/FOCS.2008.32.

3 Suman K. Bera and Amit Chakrabarti. A Depth-Five Lower Bound for Iterated Matrix Multiplication. In *30th Conference on Computational Complexity (CCC 2015)*, volume 33 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 183–197. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2015. doi:10.4230/LIPIcs.CCC.2015.183.

4 Suryajith Chillara and Partha Mukhopadhyay. On the limits of depth reduction at depth 3 over small finite fields. In *Mathematical Foundations of Computer Science (MFCS)*, pages 177–188, 2014. arXiv:1401.0189, doi:10.1007/978-3-662-44465-8_16.

5 Hervé Fournier, Nutan Limaye, Guillaume Malod, and Srikanth Srinivasan. Lower bounds for depth 4 formulas computing iterated matrix multiplication. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC 2014)*, pages 128–135, 2014. doi:10.1145/2591796.2591824.

6 Dima Grigoriev and Marek Karpinski. An exponential lower bound for depth 3 arithmetic circuits. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing (STOC 1998)*, pages 577–582, 1998. doi:10.1145/276698.276872.

7 Dima Grigoriev and Alexander A. Razborov. Exponential lower bounds for depth 3 arithmetic circuits in algebras of functions over finite fields. *Appl. Algebra Eng. Commun. Comput.*, 10(6):465–487, 2000. Preliminary version in the *39th Annual IEEE Symposium on Foundations of Computer Science (FOCS 1998)*. doi:10.1007/s002009900021.

8 Ankit Gupta, Pritish Kamath, Neeraj Kayal, and Ramprasad Saptharishi. Arithmetic Circuits: A Chasm at Depth Three. In *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2013)*, pages 578–587, 2013. doi:10.1109/FOCS.2013.68.

9 Ankit Gupta, Pritish Kamath, Neeraj Kayal, and Ramprasad Saptharishi. Approaching the chasm at depth four. *Journal of the ACM*, 61(6):33:1–33:16, 2014. Preliminary version in the *28th Annual IEEE Conference on Computational Complexity (CCC 2013)*. doi:10.1145/2629541.

10 Laurent Hyafil. On the parallel evaluation of multivariate polynomials. *SIAM Journal of Computing*, 8(2):120–123, 1979. Preliminary version in the *10th Annual ACM Symposium on Theory of Computing (STOC 1978)*. doi:10.1137/0208010.

11 Kyriakos Kalorkoti. A Lower Bound for the Formula Size of Rational Functions. *SIAM Journal of Computing*, 14(3):678–687, 1985. doi:10.1137/0214050.

12 Neeraj Kayal. An exponential lower bound for the sum of powers of bounded degree polynomials. In *Electronic Colloquium on Computational Complexity (ECCC)TR12-081*, 2012. URL: http://eccc.hpi-web.de/report/2012/081/.

13 Neeraj Kayal, Nutan Limaye, Chandan Saha, and Srikanth Srinivasan. An Exponential Lower Bound for Homogeneous Depth Four Arithmetic Circuits. In *Proceedings of the 55th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2014)*, 2014. doi:10.1109/FOCS.2014.15.

14 Neeraj Kayal and Chandan Saha. Lower bounds for depth three arithmetic circuits with small bottom fanin. In *30th Conference on Computational Complexity (CCC 2015)*, volume 33 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 158–208. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2015. doi:10.4230/LIPIcs.CCC.2015.158.

15 Neeraj Kayal, Chandan Saha, and Ramprasad Saptharishi. A super-polynomial lower bound for regular arithmetic formulas. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC 2014)*, pages 146–153, 2014. doi:10.1145/2591796.2591847.

16 Pascal Koiran. Arithmetic circuits: The chasm at depth four gets wider. *Theoretical Computer Science*, 448:56–65, 2012. arXiv:1006.4700, doi:10.1016/j.tcs.2012.03.041.

17 Mrinal Kumar and Ramprasad Saptharishi. Finer Separations Between Shallow Arithmetic Circuits. In *36th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2016)*, volume 65 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 38:1–38:12. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.FSTTCS.2016.38.

18 Mrinal Kumar and Shubhangi Saraf. The limits of depth reduction for arithmetic formulas: it's all about the top fan-in. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC 2014)*, pages 136–145, 2014. doi:10.1145/2591796.2591827.

19 Mrinal Kumar and Shubhangi Saraf. On the power of homogeneous depth 4 arithmetic circuits. In *Proceedings of the 55th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2014)*, 2014. doi:10.1109/FOCS.2014.46.

20 Noam Nisan and Avi Wigderson. Lower bounds on arithmetic circuits via partial derivatives. *Computational Complexity*, 6(3):217–234, 1997. Available on citeseer:10.1.1.90.2644. doi:10.1007/BF01294256.

21 Ran Raz. Tensor-rank and lower bounds for arithmetic formulas. In *Proceedings of the 42nd Annual ACM Symposium on Theory of Computing (STOC 2010)*, pages 659–666, 2010. doi:10.1145/2535928.

22 Alexander A. Razborov. Lower bounds on the size of bounded depth circuits over a complete basis with logical addition. *Mathematical notes of the Academy of Sciences of the USSR*, 41(4):333–338, 1987. doi:10.1007/BF01137685.

23 Ramprasad Saptharishi. A survey of lower bounds in arithmetic circuit complexity. Github survey, 2015. URL: https://github.com/dasarpmar/lowerbounds-survey/releases/.

24 Roman Smolensky. Algebraic methods in the theory of lower bounds for boolean circuit complexity. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing (STOC 1987)*, pages 77–82, 1987. doi:10.1145/28395.28404.

25 Sébastien Tavenas. Improved bounds for reduction to depth 4 and depth 3. *Inf. Comput.*, 240:2–11, 2015. Preliminary version in the *38th Internationl Symposium on the Mathematical Foundations of Computer Science (MFCS 2013)*. doi:10.1016/j.ic.2014.09.004.

26 Leslie G. Valiant, Sven Skyum, S. Berkowitz, and Charles Rackoff. Fast Parallel Computation of Polynomials Using Few Processors. *SIAM Journal of Computing*, 12(4):641–644, 1983. Preliminary version in the *6th Internationl Symposium on the Mathematical Foundations of Computer Science (MFCS 1981)*. doi:10.1137/0212043.

27 Marc van Leeuwen. https://math.stackexchange.com/questions/298836/sylvester-rank-inequality-operatornamerank-a-operatornamerankb-leq-o/298944#298944.

A Tight analysis of the [19] lower bound

We recall the measure of *projected shifted partial derivatives* that was used in [13] and [19].

$$\Gamma^{\mathrm{PSD}}_{k,\ell}(P) \quad = \quad \mathsf{Dim}\left\{\mathrm{mult}\left(\mathbf{x}^{=\ell}\partial^{=k}(P)\right)\right\}$$

where $\mathrm{mult}(f)$ is just the polynomial f restricted to just its multilinear monomials. As mentioned before, this $\Gamma^{\mathrm{PSD}}_{k,\ell}(P)$ is precisely $\mathsf{Rank}(C'_{k,\ell}(P))$ as defined in Section 6.1.

The goal of this section would be to prove Lemma 18 that we restate below.

▶ **Lemma 18** (restated). *For every d and $k = O(\sqrt{d})$ there exists parameters m, e, ϵ such that $m = \Theta(d^2)$, $n = md$ and $\epsilon = \Theta\left(\frac{\log d}{\sqrt{d}}\right)$ with*

$$\begin{aligned} m^k &\geq (1+\epsilon)^{2(d-k)} \\ m^{e-k} &= \left(\frac{2}{1+\epsilon}\right)^{d-k} \cdot \mathrm{poly}(m). \end{aligned}$$

For any $\{d, m, e, k, \epsilon\}$ satisfying the above constraints, the polynomial $\mathsf{NW}_{d,m,e}$, if $\ell = \frac{n}{2}(1-\epsilon)$, then over any field $\mathbb{F}$, we have

$$\Gamma^{\mathrm{PSD}}_{k,\ell}(\mathsf{NW}_{d,m,e}) \quad \geq \quad \binom{n}{\ell + d - k} \cdot \exp(-O(\log^2 d)).$$

The rest of this section would just be a proof of this lemma.

Before we proceed to the lower bound on $\Gamma^{\mathrm{PSD}}_{k,\ell}(\mathsf{NW}_{d,m,e})$, let us first show that we can indeed find parameters that satisfy the above constraints. Fix m to be the smallest power of 2 greater than d^2 to get $m = \Theta(d^2)$. Next, we shall fix the constant c in $\epsilon = \frac{\log d}{c\sqrt{d}}$ so that

$$m^k \quad \geq \quad (1+\epsilon)^{2(d-k)}.$$

This is always possible by choosing c to be large enough as $(1+\epsilon)^{d-k} = \exp(O(\sqrt{d}\log d))$ and that is also the order of m^k.

Once we have done that, we shall fix e so as to ensure that

$$m^{e-k} \quad = \quad \left(\frac{2}{1+\epsilon}\right)^{d-k} \cdot \mathrm{poly}(m),$$

which can be done by setting

$$e = \left\lceil \left(\frac{d-k}{\log m}\right) \cdot \log\left(\frac{2}{1+\epsilon}\right) + k \right\rceil.$$

Since changing e by 1 changes the LHS by a factor of m, the above choice would ensure the LHS and RHS are within a multiplicative factor of m. Note that this definition of e also ensures that $e \leq d - k$.

All lower bounds on the dimension of shifted partial derivatives of a polynomial P was obtained by finding a *large* set of *distinct leading monomials*. In [19], they take this approach but require a very careful analysis. The key difference in this setting is the following:

> If β is the leading monomial of a polynomial P, then for any monomial μ, we also have that $\gamma = \beta \cdot \mu$ is the leading monomial of μP.
> However, the leading monomial of $\mathrm{mult}(\mu P)$ could be $\beta' \cdot \mu$ for some $\beta' \neq \beta$ (as higher monomials could be made non-multilinear during the shift by μ).

The multilinear projection makes the task of counting leading monomials much harder and [19] come up with an indirect way to count them. Throughout this discussion, let $\mathrm{LM}(f)$ refer to the leading monomial of f in some natural ordering, say the lexicographic order.

Leading monomials after multilinear projections

Let P the polynomial of degree d for which we are trying to lower bound $\Gamma^{\mathrm{PSD}}_{k,\ell}(P)$. For every monomial multilinear monomial α of degree k, and a monomial $\beta \in \partial_\alpha(P)$, define the set $A(\alpha, \beta)$ as

$$A(\alpha,\beta) \quad = \quad \left\{ \gamma \ : \ \begin{array}{c} \mathsf{Deg}(\gamma) = \ell + d - k \text{ and there is a } mu \text{ of degree } \ell \\ \text{such that } \gamma = \mathrm{LM}(\mathrm{mult}(\mu \cdot \partial_\alpha(P))) = \mu \cdot \beta \end{array} \right\}.$$

In other words, these are the set of leading monomials obtained from a specific monomial β in $\partial_\alpha(P)$. We then have

$$\Gamma^{\mathrm{PSD}}_{k,\ell}(P) \quad \geq \quad \left| \bigcup_{\alpha,\beta} A(\alpha,\beta) \right|.$$

Choice of derivatives: Instead of looking at all derivatives in $\partial^{=k}$, we shall restrict ourselves to just a subset of derivatives. Restricting the above union to a subset $\Delta \subset \mathbf{x}^{=k}$ still continues to remain a lower bound for $\Gamma^{\mathrm{PSD}}_{k,\ell}(P)$. Keeping in mind that we are dealing with $P = \mathsf{NW}_{d,m,e}$ and that $m^k > (1+\epsilon)^{2(d-k)}$. We shall choose Δ to be a set of size exactly $(1+\epsilon)^{2(d-k)}$ which consists of monomials of the form $x_{1a_1} \cdots x_{ka_k}$ with each $a_i \leq m$. This shall become relevant later.

$$\Gamma^{\mathrm{PSD}}_{k,\ell}(P) \quad \geq \quad \left| \bigcup_{\substack{\alpha \in \Delta \\ \beta \in \partial_\alpha(P)}} A(\alpha,\beta) \right|. \tag{A.1}$$

We are abusing notation here to use $\beta \in \partial_\alpha(P)$ to mean that β is a non-zero monomial in $\partial_\alpha(P)$.

We shall need the following lemma from [19] that is a strengthening of the standard Inclusion-Exclusion principle.

▶ **Lemma 28** (Stronger Inclusion-Exclusion [19]). *Let $A_1, \ldots, A_r$ be sets and suppose*

$$\lambda := \frac{\sum_{i \neq j} |A_i \cap A_j|}{\sum_i |A_i|} \geq 1.$$

Then,

$$\left| \bigcup_i A_i \right| \quad \geq \quad \left(\frac{1}{4\lambda}\right) \cdot \left(\sum_i |A_i| \right).$$

▶ **Corollary 29.** *Considers sets $A_1, \ldots, A_r$ and let $S_1 = \sum_i |A_i|$ and $S_2 = \sum_{i \neq j} |A_i \cap A_j|$. Then,*

$$\left| \bigcup A_i \right| \quad \geq \quad \frac{S_1}{4} \cdot \min\left(1, \frac{S_1}{S_2}\right).$$

Estimating $|\bigcup A(\alpha,\beta)|$ via Inclusion-Exclusion

$$\left| \bigcup_{\alpha,\beta} A(\alpha,\beta) \right| \quad \geq \quad \sum_{\alpha,\beta} |A(\alpha,\beta)| \quad - \quad \sum_{(\alpha,\beta) \neq (\alpha',\beta')} |A(\alpha,\beta) \cap A(\alpha',\beta')|.$$

Let us first address the term $\sum |A(\alpha,\beta)|$. As mentioned earlier, it is not an easy task to get a good handle on the set $A(\alpha,\beta)$ for polynomial such as NW, for any reasonable monomial ordering. However, [19] circumvent this difficult by using an indirect approach to estimate this term.

For any derivative α and $\beta \in \partial_\alpha(P)$, define the set $S(\alpha,\beta)$ as the following set of multilinear monomials of degree ℓ that is disjoint from β.

$$S(\alpha,\beta) \quad = \quad \left\{ \mu \ : \ \begin{array}{c} \mu \text{ is multilinear, has} \\ \text{degree } \ell \text{ and } \gcd(\beta,\mu) = 1 \end{array} \right\}.$$

This on the other hand is independent of any monomial ordering, and is also easy to calculate:

$$\text{For every } \alpha,\beta \qquad |S(\alpha,\beta)| \quad = \quad \binom{n-d+k}{\ell}.$$

▶ **Lemma 30** ([19]). *For any α,*

$$\sum_{\beta} |A(\alpha,\beta)| \quad \geq \quad \left| \bigcup_{\beta} S(\alpha,\beta) \right|.$$

Proof. Consider any $\mu \in \bigcup_\beta S(\alpha,\beta)$. By definition, there is at least one multilinear monomial in $\mu \cdot \partial_\alpha(P)$. Thus, in particular LM(mult$(\mu \cdot \partial_\alpha(P))$ is non-zero and equal to some $\mu \cdot \beta$ for some monomial $\beta \in \partial_\alpha(P)$. This also implies that $\gamma = \mu \cdot \beta \in A(\alpha,\beta)$. This yields an injective map ϕ

$$\phi : \bigcup_{\beta} S(\alpha,\beta) \quad \rightarrowtail \quad \{(\beta,\gamma) \ : \ \beta \in \partial_\alpha(P) \ , \ \gamma \in A(\alpha,\beta)\}.$$

Since the size of the RHS is precisely $\sum_\beta |A(\alpha,\beta)|$, the lemma follows. ◀

Thus, by another use of Inclusion-Exclusion on the $S(\alpha,\beta)$'s, we get

$$\begin{aligned}
\left| \bigcup_{\alpha,\beta} A(\alpha,\beta) \right| \quad &\geq \quad \sum_{\alpha,\beta} |A(\alpha,\beta)| \quad - \quad \sum_{(\alpha,\beta)\neq(\alpha',\beta')} |A(\alpha,\beta) \cap A(\alpha',\beta')| \\
&\geq \quad \sum_{\alpha} \left(\sum_{\beta} |S(\alpha,\beta)| \right) \quad - \quad \sum_{\alpha} \left(\sum_{\beta\neq\beta'} |S(\alpha,\beta) \cap S(\alpha,\beta')| \right) \\
&\qquad - \quad \sum_{(\alpha,\beta)\neq(\alpha',\beta')} |A(\alpha,\beta) \cap A(\alpha',\beta')|.
\end{aligned}$$

Let us call the three terms in the RHS of the last equation as T_1, T_2 and T_3 respectively. Since we know the size of each $S(\alpha,\beta)$ exactly, the value of T_1 is easily obtained.

▶ **Lemma 31** ([19]).

$$T_1(\alpha) \quad := \quad \sum_{\beta} |S(\alpha,\beta)| \quad = \quad (\#\ \textit{mons in a deriv}) \cdot \binom{n-d+k}{\ell}.$$

We shall be simplifying such binomial coefficients very often.

▶ **Lemma 32.** *Let n and ℓ be parameters such that $\ell = \frac{n}{2}(1-\epsilon)$ for some $\epsilon = o(1)$. For any a, b such that $|a|, |b| = O(\sqrt{n})$,*

$$\binom{n-a}{\ell-b} \quad = \quad \binom{n}{\ell} \cdot 2^{-a} \cdot (1+\epsilon)^{a-2b} \cdot \exp(-O(b \cdot \epsilon^2)).$$

Proof. The proof of the above lemma would repeatedly use the fact that $n! = (n-a)! \cdot n^a \cdot \text{poly}(n)$ whenever $a = O(\sqrt{n})$ (see [9, Lemma 3.4]).

$$\begin{aligned}
\frac{\binom{n-a}{\ell-b}}{\binom{n}{\ell}} \quad &= \quad \frac{(n-a)!}{n!} \cdot \frac{\ell!}{(\ell-b)!} \cdot \frac{(n-\ell)!}{(n-\ell-a+b)!} \\
&\overset{\text{poly}}{\approx} \quad \frac{1}{n^a} \cdot \ell^b \cdot \frac{(n-\ell)^a}{(n-\ell)^b} \\
&= \quad \frac{\left(\frac{n}{2}\right)^a (1+\epsilon)^a}{n^a} \cdot \frac{(1-\epsilon)^b}{(1+\epsilon)^b} \\
&= \quad 2^{-a} \cdot (1+\epsilon)^{a-2b} \cdot \exp(-O(b \cdot \epsilon^2)).
\end{aligned}$$

◀

Since our of parameters would be $\epsilon = \Theta\left(\frac{\log d}{\sqrt{d}}\right)$, the bound on T_1 can be simplified as

$$\begin{aligned} T_1(\alpha) &= (\#\text{ mons in a deriv}) \cdot \binom{n}{\ell} \cdot \left(\frac{1+\epsilon}{2}\right)^{d-k} \cdot O(1) \\ &= m^{e-k} \cdot \binom{n}{\ell} \cdot \left(\frac{1+\epsilon}{2}\right)^{d-k} \cdot O(1) \\ &= \binom{n}{\ell} \cdot O(1). \end{aligned}$$

where we used the fact that every non-zero k-th order derivative of $\mathsf{NW}_{d,m,e}$ has exactly m^{e-k} monomials and our setting of parameters.

▶ Remark. In this particular instance, the approximation factor was just $O(1)$ but we would often have to deal with situations where this factor is $\exp(O(\log^2 d))$. To avoid writing this factor of $\exp(O(\log^2 d))$, we shall use $\approx$ of $\gtrsim$ or $\lesssim$ to indicate that a factor $\exp(O(\log^2 d))$ is omitted.

We now move on to the calculation of T_2. This is the first place where the choice of the polynomial and parameters becomes crucial.

▶ **Lemma 33** ([19]). *For the polynomial $P = \mathsf{NW}_{d,m,e}$, if $n = md$ and $\ell = \frac{n}{2}(1-\epsilon)$ for $\epsilon = \Theta\left(\frac{\log d}{\sqrt{d}}\right)$, for any $\alpha \in \Delta$*

$$T_2(\alpha) := \sum_{\beta \neq \beta'} |S(\alpha,\beta) \cap S(\alpha,\beta')| \lesssim m^{2(e-k)} \cdot \binom{n}{\ell} \cdot \left(\frac{1+\epsilon}{2}\right)^{2d-2k}.$$

Proof. Recall that $S(\alpha,\beta) \cap S(\alpha,\beta')$ is just set of all multilinear monomials μ of degree ℓ that are disjoint from both β and β'. Hence, for any pair of multilinear degree $(d-k)$ monomials $\beta \neq \beta' \in \partial_\alpha(P)$ such that $\mathsf{Deg}(\gcd(\beta,\beta')) = t$,

$$|S(\alpha,\beta) \cap S(\alpha,\beta')| = \binom{n-2d+2k+t}{\ell}.$$

Thus, if we can count the number of pairs (β,β') that agree on exactly t places, we can obtain $T_2(\alpha)$. Note that for $\mathsf{NW}_{d,m,e}$, any two $\beta,\beta' \in \partial_\alpha(\mathsf{NW}_{d,m,e})$ can agree on at most $e-k$ places. Further, the number of pairs that agree in exactly $0 \le t \le e-k$ places is at most

$$m^{e-k} \cdot \binom{d-k}{t} \cdot (m-1)^{e-k-t}$$

as there are m^{e-k} choices for β, and $\binom{d-k}{t}$ choices for places where they may agree, and

$(m-1)^{e-k-t}$ choices for β' that agree with β on those t places. Thus,

$$\begin{aligned}
T_2(\alpha) &\leq \sum_{t=0}^{e-k} m^{e-k} \cdot \binom{d-k}{t} \cdot (m-1)^{e-k-t} \cdot \binom{n-2d+2k+t}{\ell} \\
&\approx \sum_{t=0}^{e-k} m^{e-k} \cdot \binom{d-k}{t} \cdot (m-1)^{e-k-t} \cdot \binom{n}{\ell} \frac{1}{2^{2d-2k-t}} \cdot (1+\epsilon)^{2d-2k-t} \\
&\leq m^{2(e-k)} \binom{n}{\ell} \left(\frac{1+\epsilon}{2}\right)^{2d-2k} \cdot \sum_{t=0}^{e-k} \binom{d-k}{t} \left(\frac{2}{(1+\epsilon)m}\right)^{t} \\
&\leq m^{2(e-k)} \binom{n}{\ell} \left(\frac{1+\epsilon}{2}\right)^{2d-2k} \cdot \left(1 + \frac{2}{(1+\epsilon)m}\right)^{d-k} \\
&= m^{2(e-k)} \cdot \binom{n}{\ell} \cdot \left(\frac{1+\epsilon}{2}\right)^{2d-2k} \cdot O(1) \qquad \text{as } m = \Omega(d).
\end{aligned}$$

◀

Combining this with Lemma 31 and using Inclusion-Exclusion (Corollary 29), we get that for every $\alpha \in \Delta$,

$$\begin{aligned}
\left|\bigcup_{\beta} S(\alpha,\beta)\right| &\gtrsim T_1(\alpha) \cdot \min\left(1, \frac{T_1(\alpha)}{T_2(\alpha)}\right) \\
&\approx T_1(\alpha) \cdot \min\left(1, \frac{\left(\frac{2}{1+\epsilon}\right)^{d-k}}{m^{e-k}}\right) \\
&\approx T_1(\alpha).
\end{aligned}$$

by our choice of parameters. Note that e needs to tailored very precisely to force the above condition! If e is chosen too large or small, we get nothing from this whole exercise!

Thus by Lemma 30 and Lemma 31, we get

$$\sum_{\substack{\alpha\in\Delta \\ \beta\in\partial_\alpha(P)}} |A(\alpha,\beta)| \geq |\Delta| \cdot \left|\bigcup_{\beta} S(\alpha,\beta)\right| \geq |\Delta| \cdot T_1(\alpha) \approx |\Delta| \cdot \binom{n}{\ell}. \tag{A.2}$$

Upper bounding $\sum |A(\alpha,\beta) \cap A(\alpha',\beta')|$

We are still left with the task of upper bounding

$$T_3 = \sum_{(\alpha,\beta)\neq(\alpha',\beta')} |A(\alpha,\beta) \cap A(\alpha',\beta')|.$$

As mentioned earlier, we really do not have a good handle on the set $A(\alpha,\beta)$, and certainly not on the intersection of two such sets. Once again, we shall use a proxy that is easier to estimate to upper bound T_3.

The set $A(\alpha,\beta) \cap A(\alpha',\beta')$ consists of multilinear monomials γ of degree $\ell + d - k$ such that there exists multilinear monomials μ, μ' of degree ℓ satisfying

$$\begin{aligned}
\gamma &= \mu\beta = \mu'\beta', \\
\mu\beta &= \mathrm{LM}(\mathrm{mult}(\mu\partial_\alpha(P))) \\
\text{and } \mu'\beta' &= \mathrm{LM}(\mathrm{mult}(\mu'\partial_{\alpha'}(P))).
\end{aligned}$$

This in particular implies that γ must be divisible by both β and β'.

▶ **Observation 34.** *If* $\mathsf{Deg}(\gcd(\beta,\beta')) = t$*, then*

$$|A(\alpha,\beta) \cap A(\alpha',\beta')| \quad \leq \quad \binom{n-2d+2k+t}{\ell-d+k+t}.$$

Proof. Every monomial $\gamma \in A(\alpha,\beta) \cap A(\alpha',\beta')$ must be divisible by β and β'. Since $|\beta \cup \beta'| = 2d-2k-t$, the number of choices of γ is precisely

$$\binom{n-(2d-2k-t)}{(\ell+d-k)-(2d-2k-t)} \quad = \quad \binom{n-2d+2k+t}{\ell-d+k+t}. \qquad \blacktriangleleft$$

One needs a similar argument as in the case of T_2 to figure out how many pairs $(\alpha,\beta) \neq (\alpha',\beta')$ are there with $\mathsf{Deg}(\gcd(\beta,\beta')) = t$ and sum them up accordingly.

▶ **Lemma 35** ([19])**.** *For the polynomial* $\mathsf{NW}_{d,m,e}$*, and* $n = md$ *and* $\ell = \frac{n}{2}(1-\epsilon)$ *for* $\epsilon = \Theta\left(\frac{\log d}{\sqrt{d}}\right)$*,*

$$\sum_{(\alpha,\beta)\neq(\alpha',\beta')} |A(\alpha,\beta) \cap A(\alpha',\beta')| \quad \lesssim \quad |\Delta|^2 \cdot \left(\frac{m^{e-k}}{2^{d-k}}\right)^2 \cdot \binom{n}{\ell}.$$

Proof. Fix a pair of derivatives α, α'. Let

$$T_3(\alpha,\alpha') \quad := \quad \sum_{\substack{\beta\in\partial_\alpha(P)\\ \beta'\in\partial_{\alpha'}(P)\\ (\alpha,\beta)\neq(\alpha',\beta')}} |A(\alpha,\beta) \cap A(\alpha',\beta')|\,.$$

As before, we shall first count the number of pairs of monomials $\beta \in \partial_\alpha P$ and $\beta' \in \partial_{\alpha'} P$ such that $\gcd(\beta,\beta') = t$. Note that since α may differ from α', we could potentially have $\gcd(\beta_1,\beta_2) = e$. Once again, this is easily seen to be at most

$$m^{e-k} \cdot \binom{d-k}{t} \cdot (m-1)^{e-k-t}.$$

Therefore, using Observation 34,

$$\begin{aligned}
T_3(\alpha,\alpha') \quad &\leq \quad \sum_{t=0}^{e} m^{e-k} \cdot (m-1)^{e-k-t} \binom{d-k}{t} \binom{n-2d+2k+t}{\ell-d+k+t} \\
&\approx \quad \sum_{t=0}^{e} m^{e-k} \cdot (m-1)^{e-k-t} \binom{d-k}{t} \cdot \binom{n}{\ell} \left(\frac{1}{2}\right)^{2d-2k-t} (1+\epsilon)^t \\
&\leq \quad \frac{m^{2(e-k)}}{2^{2(d-k)}} \cdot \binom{n}{\ell} \cdot \left(1+\frac{2(1+\epsilon)}{m}\right)^{d-k} \\
&\approx \quad \left(\frac{m^{e-k}}{2^{d-k}}\right)^2 \cdot \binom{n}{\ell}. \\
\implies T_3 \quad &\lesssim \quad |\Delta|^2 \cdot \left(\frac{m^{e-k}}{2^{d-k}}\right)^2 \cdot \binom{n}{\ell}.
\end{aligned}$$

◀

Recalling that we have chosen our parameters so that

$$\frac{m^{e-k}}{2^{d-k}} \approx \left(\frac{1}{1+\epsilon}\right)^{d-k} \quad \text{and} \quad |\Delta| = (1+\epsilon)^{2(d-k)},$$

the above equation reduces to

$$T_3 = \sum_{(\alpha,\beta)\neq(\alpha',\beta')} |A(\alpha,\beta) \cap A(\alpha',\beta')| \lesssim |\Delta| \cdot \binom{n}{\ell}.$$

Combining with (A.2), we obtain the required bound for $|\bigcup A(\alpha,\beta)|$ via Inclusion-Exclusion (Corollary 29).

$$\Gamma^{\mathrm{PSD}}_{k,\ell}(\mathsf{NW}_{d,m,e}) \geq \left|\bigcup_{\alpha,\beta} A(\alpha,\beta)\right| \gtrsim \binom{n}{\ell} \cdot (1+\epsilon)^{2d-2k}.$$

The only thing left to observe is that by Lemma 32,

$$\binom{n}{\ell+d-k} \approx \binom{n}{\ell} \cdot (1+\epsilon)^{2d-2k},$$

and that completes the proof of Lemma 18. ◀

Complete Derandomization of Identity Testing and Reconstruction of Read-Once Formulas*

Daniel Minahan[1] and Ilya Volkovich[2]

1 **Departments of Mathematics and EECS, CSE Division, University of Michigan, Ann Arbor, MI, USA**
dminahan@umich.edu

2 **Department of EECS, CSE Division, University of Michigan, Ann Arbor, MI, USA**
ilyavol@umich.edu

Abstract

In this paper we study the identity testing problem of *arithmetic read-once formulas* (ROF) and some related models. A read-once formula is formula (a circuit whose underlying graph is a tree) in which the operations are $\{+, \times\}$ and such that every input variable labels at most one leaf. We obtain the first polynomial-time deterministic identity testing algorithm that operates in the black-box setting for read-once formulas, as well as some other related models. As an application, we obtain the first polynomial-time deterministic reconstruction algorithm for such formulas. Our results are obtained by improving and extending the analysis of the algorithm of [52].

1998 ACM Subject Classification F.2.0 Analysis of Algorithms and Problem Complexity

Keywords and phrases Derandomization, Read-Once Formulas, Identity Testing, Arithmetic Circuits, Reconstruction

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.32

1 Introduction

In this paper we study the problem of Polynomial Identity Testing (PIT): given an arithmetic circuit C over a field $\mathbb{F}$, with input variables $x_1, x_2, \ldots, x_n$, determine whether C computes the identically zero polynomial. Given its connections to a wide range of problems, PIT is considered a central problem in algebraic complexity theory and algorithms design. Particular instances include: perfect matchings in graphs [40, 43, 20], primality testing [2], $\mathsf{IP} = \mathsf{PSPACE}$ [41, 48] the PCP theorem [10, 9] and many more. PIT is one of a few natural problems which have a simple efficient randomized algorithm [18, 47, 55] but lack a deterministic one. Indeed, it has been a long standing open question to come up with an efficient deterministic algorithm for this problem.

In this paper we consider the PIT problem in the *black-box* setting. In this setting, one is not given the full description of the circuit C but only allowed black-box (oracle) access to C. The problem of derandomizing identity testing in this setting reduces to that of finding for every s an explicit set of points $\mathcal{H} \subseteq \mathbb{F}^n$ of size $\mathrm{poly}(s)$ such that any non-zero circuit of size s does not vanish on $\mathcal{H}$. We refer to such sets as *hitting sets*. Indeed, the randomized algorithm of [18, 47, 55] provides an exponential-size hitting set. Furthermore, applying

* Research partially supported by NSF.

COMPUTATIONAL COMPLEXITY CONFERENCE

standard probabilistic arguments one can show existence of "small" hitting sets. Yet, coming up with an explicit hitting set is believed to be very difficult task as it would immediately imply explicit exponential lowers bounds [30, 1].

Yet, for several restricted classes of arithmetic circuits, efficient deterministic black-box PIT algorithms were found. For example, efficient black-box PIT algorithms were shown for depth-2 arithmetic circuits [13, 36, 39] and depth-3 arithmetic circuits with bounded top fan-in (also known as $\Sigma\Pi\Sigma(k)$ circuits) [19, 35, 34, 11, 45, 33, 46, 3]. There has also been a lot of progress on PIT for restricted classes of depth-4 circuits [44, 11, 12, 3, 32, 26, 7, 42, 52, 37, 38]. Another body of research has been focused on PIT algorithms for bounded-read models. That is, classes of circuits where each variable appears some bounded number of times [22, 4, 24, 23, 21, 7, 52, 28, 6, 27, 20], with the simplest case being the read-once formulas.

A *read-once formula* (ROF for short) is an arithmetic formula (i.e. a tree) in which the operations are $\{+, \times\}$ and such that every input variable labels at most one leaf. These formulas can be thought of as the smallest formulas that depend on all their variables and the simplest non-trivial subclass of multilinear formulas. Although ROFs form a very restricted model of computation they have received a lot of attention in both the Boolean [31, 8, 17] and the algebraic [29, 16, 14, 15, 51, 52, 54] worlds.

While ROFs have a trivial polynomial (and, in fact, linear-time) white-box PIT algorithm, the first sub-exponential time $n^{\mathcal{O}(\sqrt{n})}$ black-box PIT algorithm for ROFs was given in [49]. Later on, in [52][1] the result was improved to $n^{\mathcal{O}(\log n)}$ via another algorithm. A different analysis for the latter algorithm, resulting in roughly the same run time, was given in [7]. Yet, despite the rich body of work devoted to the problem, prior to our work no polynomial-time black-box PIT algorithm was known even for ROFs. In this paper we give the first black-box PIT algorithm for ROFs, and some related classes of formulas, thus achieving a complete derandomization of the PIT problem for these classes. For more information on PIT we refer the reader to the survey [53].

It is important to point out that while PIT asks whether the resulting polynomial is identically zero as a formal sum of monomials, some non-identically zero polynomials might evaluate to the zero function. For example, $x^5 - x$ will always evaluate to zero over the field of five elements. For this reason we will allow our algorithm to evaluate the polynomial on elements from a polynomially large extension field of $\mathbb{F}$. In [52] it was shown one cannot achieve polynomial-time black-box PIT algorithms if $|\mathbb{F}| = o(n/\log n)$.

1.1 Our Results

In this section we describe and discuss our results.. In fact, our results hold for the slightly richer class of preprocessed read-once formulas. A *preprocessed ROF* (PROF for short) is a ROF in which we are allowed to replace each variable x_i with a univariate polynomial $T_i(x_i)$. A polynomial $P(\bar{x})$ is a *Preprocessed Read-Once Polynomial* (PROP for short) if it can be computed by a preprocessed read-once formula. This PROPs also generalize the "sum-of-univariates" model. (see Section 3.2 for a formal definition). We begin with our main result: polynomial-time black-box PIT algorithm for PROFs.

▶ **Theorem 1.** *Let $n, d \in \mathbb{N}$. There exist a deterministic algorithm that given black-box (oracle) access to a preprocessed read-once formula Φ on n variables and individual degrees (of the preprocessing) at most d, checks whether $\Phi \equiv 0$. The running time of the algorithm is polynomial in n and d.*

[1] Conference version first appeared in [50].

In [52] it was shown how to extend a PIT algorithm for a single PROF into a PIT algorithm for a sum of PROFs. By plugging in our main result we obtain a black-box PIT algorithm for sums of PROFs.

▶ **Theorem 2.** *Let $k, n, d \in \mathbb{N}$. There exist a deterministic algorithm that given black-box (oracle) access to $\Phi = \Phi_1 + \ldots + \Phi_k$, where the Φ_i-s are preprocessed read-once formulas in n variables, with individual degrees at most d, checks whether $\Phi \equiv 0$. The running time of the algorithm is $(nd)^{\mathcal{O}(k)}$.*

Observe that for a fixed $k \in \mathbb{N}$ the algorithm runs in polynomial time with respect to n and d. Furthermore, observe that if $\mathcal{H}$ is hitting set for a sum of two PROPs then $\mathcal{H}$ is an *interpolating* set for a single PROP. That is, the values of a single PROP P on $\mathcal{H}$ contain enough information to uniquely identify P. Indeed, a consequence of Theorem 2 is an interpolating set of polynomial size for PROPs. However in general, $\mathcal{H}$ does not provide us with an efficient algorithm to reconstruct a corresponding PROF.

In [16], a randomized polynomial-time reconstruction algorithm for ROFs was given. In [51], the algorithm was extended to PROFs. Moreover, it was shown how to convert a black-box PIT algorithm into a reconstruction algorithm paying a polynomial overhead. Indeed, by plugging in the result of [52], the first deterministic sub-exponential (and, in fact, quasi-polynomial) time reconstruction algorithm for PROFs was given. By plugging in our main result, we achieve a complete derandomization of the reconstruction algorithm by obtaining a deterministic polynomial-time reconstruction algorithm for PROFs.

▶ **Theorem 3.** *There exist a deterministic algorithm that given black-box (oracle) access to a preprocessed read-one formula Φ, on n variables and individual degrees at most d, reconstructs Φ. Namely, the algorithm outputs a PROF $\hat{\Phi}$ that computes the same polynomial. The running time of the algorithm is polynomial in n and d.*

1.2 Organization

The paper is organized as follows. In Section 2 we give the basic definitions and notations. In Section 3 we formally introduce ROFs and its generalizations along with some structural properties, and in Section 3.3 we discuss the PIT algorithm of [52]. In Section 3.4 we prove some additional properties of the algorithm, which is the main technical contribution of the paper. Next, in Section 4 we give our main result, thus proving Theorem 1. We discuss the applications of our main result in Section 5 proving Theorems 2 and 3. We conclude the paper with some open questions in Section 6.

2 Preliminaries

For a positive integer n, let $[n]$ denote the set $\{1, \ldots, n\}$. We now give some definitions that apply to polynomials $P, Q \in \mathbb{F}[x_1, \ldots, x_n]$. For a polynomial P, a variable x_i, and $\alpha \in \mathbb{F}$, let $P|_{x_i=\alpha}$ denote the polynomial that results upon setting $x_i = \alpha$. We say that P *depends* on x_i if there exist $\bar{a}, \bar{b} \in \mathbb{F}^n$ that differ in only the i-coordinate such that $P(\bar{a}) \neq P(\bar{b})$. We denote $\mathrm{var}(P) \stackrel{\Delta}{=} \{i \ : \ P \text{ depends on } x_i\}$. Intuitively, P depends on x_i if x_i appears when P is written as a sum of monomials.

We say that P is a *homogeneous* polynomial if all monomials in P has the same total degree. For $i \in \mathbb{N}$ we define $H_i[P]$ as the *homogeneous part* of degree i of P. That is, all the monomials of total degree i that appear in P. If P does not have monomials of degree i then

$H_i[P] \equiv 0$. We say that P and Q are *similar* and denote $P \sim Q$ if there exist $\alpha \in \mathbb{F} \setminus \{0\}$ such that $\alpha \cdot P = Q$.

In order to actually calculate the complexity of our algorithm we need to define a formal model of computation for polynomials.

▶ **Definition 4** (Arithmetic formula)**.** An arithmetic formula is a binary tree where each leaf is labeled with a variable $x_i \in \{x_1, \ldots, x_n\}$ and each internal node, called a gate, is labeled with an operation $+$ or $\times$. Additionally, each leaf and node are labeled with some $(\alpha, \beta) \in \mathbb{F}^2$. The tree is evaluated by recursively calculating the values of the left subtree P_1 and the right subtree P_2 and then combining them by $\alpha(P_1 \text{ op } P_2) + \beta$ where op is the operation at the top gate.

The efficiency of a formula over a set of formulas $\mathcal{C}$ is measured by the number of gates in $\mathcal{C}$. Thus when we say polynomial time, we mean polynomial in the number of gates. Often times, we will implicitly associate a class of formulas $\mathcal{C}$ with the class of polynomials computed by these formulas.

We consider formulas in the *black-box* (or oracle) setting. That is, the algorithm cannot directly look at the formula and is only allowed to query the polynomial computed by the formula on $\mathbb{F}^n$. Hereafter, we assume that an evaluation query can be carried out in $\mathcal{O}(1)$ time. In case $\mathbb{F}$ is small, we allow to query the formula on a polynomially-large extension field of $\mathbb{F}$.

2.1 Generators and Hitting Sets

Our black-box PIT algorithms use the notion of *generators*. In this section, we formally define this notion, describe a few of its useful properties and give the connection to hitting sets. Intuitively, a generator $\mathcal{G}$ for a polynomial class $\mathcal{C}$, is a function that stretches t independent variables into $n \gg t$ dependent variables that can be *plugged* into any polynomial $P \in \mathcal{C}$ without causing it to vanish. Recall that a hitting set $\mathcal{H} \subseteq \mathbb{F}^n$ for a class of polynomials $\mathcal{C}$ is a set such that for any nonzero polynomial $P \in \mathcal{C}$, there exists $\bar{a} \in \mathcal{H}$, such that $P(\bar{a}) \neq 0$.

▶ **Definition 5** (Hitting Set)**.** Let $\mathcal{C}$ be a class of polynomials in $\mathbb{F}[x_1, \ldots, x_n]$. A set $\mathcal{H}$ is called a *hitting set* for $\mathcal{C}$ provided that $\forall P \in \mathcal{C}$ with $P \not\equiv 0$ we have that $P|_{\mathcal{H}} \not\equiv 0$.

This leads us to a basic algorithm for PIT.

▶ **Lemma 6.** *Let $\mathcal{C}$ be a class of polynomials in $\mathbb{F}[x_1, \ldots, x_n]$ and let $\mathcal{H}$ be a hitting set for $\mathcal{C}$. Then there exists a deterministic PIT algorithm for $\mathcal{C}$ that runs in time $\mathcal{O}(|\mathcal{H}|)$.*

The following generalization of the fundamental theorem of algebra provides hitting sets of exponential size for every polynomial. A proof can be found in [5].

▶ **Lemma 7.** *Let $P \not\equiv 0 \in \mathbb{F}[x_1, \ldots, x_n]$ and suppose the individual degree of any variable in P is bounded by some $d \in \mathbb{N}$. Pick $S \subseteq \mathbb{F}$ with $|S| > d$. Then $P|_{S^n} \not\equiv 0$.*

▶ Remark. The precondition of the lemma implies that $|\mathbb{F}| > d$. In case that $\mathbb{F}$ is small, this assumption is met by choosing elements from an appropriately large extension field of $\mathbb{F}$.

A related notion is the notion of generators. Many hitting sets are constructed by means of generators.

▶ **Definition 8** (Generator)**.** Let $\mathcal{C}$ be a class of polynomials over $\mathbb{F}$. A polynomial map $G : \mathbb{F}^t \to \mathbb{F}^n$ is a *generator* for $\mathcal{C}$ provided that $\forall P \not\equiv 0 \in \mathcal{C}$ we have $P(G) \not\equiv 0$.

Intuitively, a generator $\mathcal{G}$ for $\mathcal{C}$ is a polynomial mapping that has a hitting set for $\mathcal{C}$ in its image. More specifically, Lemma 7 allows us to convert a generator into a hitting set by observing that a polynomial composed with a polynomial map results in another polynomial. Such composition typically reduces the number of variables that the polynomial depends on, but it may increase the total degree. Thus, since the size of the hitting set produced by Lemma 7 depends on both parameters, we want to find a generator that reduces the number of variables without drastically increasing its degree.

3 Read-Once Formulas

In this section we discuss our computational model. We first consider the basic model of read-once formulas and cover some of its main properties. Then, we introduce the model of preprocessed-read-once formulas and give its corresponding properties.

3.1 Read-Once Formulas and Read-Once Polynomials

Most of the definitions that we give in this section are from [29] and [52] or some small variants. We start by formally defining the notions of a read-once formula and a read-once polynomial.

▶ **Definition 9** (Read-Once Formula)**.** A *read-once formula* (ROF) is an arithmetic formula where each variable appears at most once. A polynomial $P(\bar{x})$ is a *read-once polynomial* (ROP for short) if it can be computed by a read-once formula.

Clearly, ROPs form a subclass of multilinear polynomials. In addition, note that the number of gates in a ROF is at most twice the number of variables. This means that our complexity scales with n, so we need only be concerned about how the runtime of our algorithm scales with respect to the number of variables. Thus, our ideal efficiency for an algorithm is $n^{\mathcal{O}(1)}$. The next lemma also follows easily from the definition.

▶ **Lemma 10** (ROP Structural Lemma)**.** *Every ROP $P(\bar{x})$ that depends on at least two variables can be presented in exactly one of the following forms:*

1. $P(\bar{x}) = P_1(\bar{x}) + P_2(\bar{x})$,
2. $P(\bar{x}) = P_1(\bar{x}) \cdot P_2(\bar{x}) + c.$,

where P_1 and P_2 are non-constant, variable-disjoint ROPs and $c \in \mathbb{F}$ is a constant.

3.2 Preprocessed Read-Once Polynomials

In this section we extend the model of ROFs by allowing a *preprocessing* step of the input variables. While the basic model is read-once in its variables, the extended model can be considered as read-once in univariate polynomials. In addition, this model generalizes the "sum-of-univariates" model, which, in particular, contains the "sum-of-squares" model.

▶ **Definition 11.** A *preprocessing* is a transformation $T(\bar{x}) : \mathbb{F}^n \to \mathbb{F}^n$ of the form $T(\bar{x}) \stackrel{\Delta}{=} (T_1(x_1), T_2(x_2), \ldots, T_n(x_n))$ such that each T_i is a *non-constant univariate polynomial*.

Notice that preprocessings do not affect the PIT problem in the white-box setting as for every n-variate polynomial $P(\bar{y})$ it holds that $P(\bar{y}) \equiv 0$ if and only if $P(T(\bar{x})) \equiv 0$. We now give a formal definition and list some immediate properties.

▶ **Definition 12.** A *preprocessed arithmetic read-once formula* (PROF for short) over a field $\mathbb{F}$ in the variables $\bar{x} = (x_1, \ldots, x_n)$ is a binary tree whose leafs are labelled with non-constant univariate polynomials $T_1(x_1), T_2(x_2), \ldots, T_n(x_n)$ (all together forming a preprocessing) and whose internal nodes are labelled with the arithmetic operations $\{+, \times\}$ and with a pair of field elements $(\alpha, \beta) \in \mathbb{F}^2$. Each T_i can label at most one leaf. The computation is performed in the following way. A leaf labelled with the polynomial $T_i(x_i)$ and with (α, β) computes the polynomial $\alpha \cdot T_i(x_i) + \beta$. If a node v is labelled with the operation *op* and with (α, β), and its children compute the polynomials Φ_{v_1} and Φ_{v_2} then the polynomial computed at v is $\Phi_v = \alpha \cdot (\Phi_{v_1}\ op\ \Phi_{v_2}) + \beta$.

A polynomial $P(\bar{x})$ is a *Preprocessed Read-Once Polynomial* (PROP for short) if it can be computed by a preprocessed read-once formula. A *Decomposition* of a polynomial P is a pair $Q(\bar{z}), T(\bar{x})$ such that $P(\bar{x}) = Q(T(\bar{x}))$ when Q is a ROP and T is a preprocessing. An immediate consequence from the definition is that each PROP admits a decomposition. The following lemma is the PROPs analog of Lemma 10.

▶ **Lemma 13** (PROP Structural Lemma). *Every PROP $P(\bar{x})$ with $|\mathrm{var}(P)| \geq 2$ can be presented in one of the following forms:*
1. $P(\bar{x}) = P_1(\bar{x}) + P_2(\bar{x})$,
2. $P(\bar{x}) = P_1(\bar{x}) \cdot P_2(\bar{x}) + c$,

where P_1 and P_2 are non-constant, variable-disjoint PROPs and $c \in \mathbb{F}$ is a constant.

3.3 The Algorithm of [52]

In this paper we improve the complexity analysis of the PIT algorithm of [52]. We begin by describing their algorithm. The heart of the algorithm is a construction of polynomial map $G_{n,t}$ which is shown to be a generator for PROPs for a certain range of parameters.

As in [52], we fix a set $A = \{\alpha_1, \alpha_2, \ldots, \alpha_n\} \subseteq \mathbb{F}$ of n distinct elements. It is also assumed that in case that $\mathbb{F}$ is small, we have access to some extension field of $\mathbb{F}$ with more than n elements. As was shown in [52], this assumption is necessary in order to achieve a polynomial-time algorithm.

▶ **Definition 14** (The generator of [52]). Let $t \in \mathbb{N}$. For each $i \in [n]$ let $L_i(y)$ denote the i-th Lagrange Interpolation polynomial. Formally: $L_i(y) \triangleq \frac{\prod_{j \neq i}(y - \alpha_j)}{\prod_{j \neq i}(\alpha_i - \alpha_j)}$. That is, $L_i(y)$ is a degree $n-1$ polynomial satisfying: $L_i(\alpha_j) = 1$ when $j = i$ and $L_i(\alpha_j) = 0$ when $j \neq i$. For each $i \in [n]$, let $G_t^i(y_1, \ldots, y_t, z_1, \ldots, z_t) \triangleq \sum_{k=1}^{t} L_i(y_k) \cdot z_k$. Finally, let $G_{n,t}(y_1, \ldots, y_t, z_1, \ldots, z_t) \triangleq \big(G_t^1(y_1, \ldots, z_t), \ldots, G_t^n(y_1, \ldots, z_t)\big)$.

$G_{n,t}$ can be seen as a sum of t variable-disjoint copies of $G_{n,1}$. This can be seen as the algebraic analogue of t-wise independent bits. The main part of the analysis of the algorithm is to establish that for every $n \in \mathbb{N}$ the map $G_{n,\log n}$ is a generator for PROPs on n variables.

▶ **Lemma 15** ([52]). *Let $P \in \mathbb{F}[x_1, \ldots, x_n]$ be a non-constant PROP. Then $P(G_{n,\log n})$ is non-constant.*

The intuition behind the proof is that a PROP can be written as either a sum or a product of two variable-disjoint polynomials (Lemma 13). Hence, (at least) one of these polynomials contains at most half of the variables. The map $G_{n,t}$ allows to "move" to a smaller polynomial by "shaving" a copy of $G_{n,1}$. Finally, applying Lemma 7 one could show that if $G_{n,t}$ is a generator for a class of polynomials, then it can be converted into a relatively small hitting set for that same class.

▶ **Lemma 16.** *Let $P \in \mathbb{F}[x_1, \ldots, x_n]$ be a polynomial of degree d such that $P(G_{n,t}) \not\equiv 0$ for some $t, d \in \mathbb{N}$. Then P has a hitting set of size $(nd)^{\mathcal{O}(t)}$.*

Consequently, the result of Lemma 15 translates into a hitting set of size $(nd)^{\mathcal{O}(\log n)}$ for PROPs. We note that the generator of [52] has been used as an ingredient in some subsequent PIT algorithms (e.g. [21, 7, 6, 25]).

3.4 Our Technical Contribution

In this section we explore additional properties of the generator of [52]. The main observation is a structural property of the generator when applied to a polynomial that depends only on a "small" subset of variables. This is the main technical contribution of the paper.

Let $A = \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ be the set of elements that is used to define the generator.

▶ **Definition 17.** For $I \subseteq [n]$, define $\Phi_I(y) \stackrel{\Delta}{=} \prod_{i \in I}(y - \alpha_i)$. For notational convenience, $\Phi_\emptyset(y) \stackrel{\Delta}{=} 1$.

In order to provide some intuition for the definition, we observe that for any $i \in [n]$ we have that $L_i \sim \Phi_{[n]\setminus\{i\}}$.

▶ **Lemma 18.** *Let $P \in \mathbb{F}[x_1, \ldots, x_n]$ be a homogeneous polynomial of a total degree d and let δ be an upper bound on the individual degrees of all variables x_i in P. Then there exists a polynomial $P'(y)$ of degree at most $\delta \cdot |\mathrm{var}(P)| - d$ such that*

$$P(G_{n,1}(y,z)) = z^d \cdot \Phi^{d-\delta}_{[n]}(y) \cdot P'(y) \cdot \Phi^{\delta}_{[n]\setminus \mathrm{var}(P)}(y).$$

In particular, there exist a polynomial $P'(y)$ of degree at most $d \cdot (|\mathrm{var}(P)| - 1)$ such that

$$P(G_{n,1}(y,z)) = z^d \cdot P'(y) \cdot \Phi^{d}_{[n]\setminus \mathrm{var}(P)}(y).$$

Proof. Let $V \subseteq [n]$ and let $m(\bar{x}) = \alpha \prod_{i \in V} x_i^{e_i}$ be a monomial s.t. $\sum_{i \in V} e_i = d$ and $\forall i \in V : 0 \leq e_i \leq \delta$.

$$m\left(G_{n,1}(y,z)\right) = \alpha z^d \cdot \prod_{i \in V} L_i^{e_i}(y) = \beta z^d \cdot \prod_{i \in V} \Phi^{e_i}_{[n]\setminus\{i\}}(y) = \beta z^d \cdot \Phi^d_{[n]}(y) / \prod_{i \in V}(y - \alpha_i)^{e_i} =$$
$$\beta z^d \cdot \Phi^{d-\delta}_{[n]}(y) \cdot \Phi^{\delta}_{V}(y) / \prod_{i \in V}(y - \alpha_i)^{e_i} \cdot \Phi^{\delta}_{[n]\setminus V}(y) =$$
$$z^d \cdot \Phi^{d-\delta}_{[n]}(y) \cdot \beta \prod_{i \in V}(y - \alpha_i)^{\delta - e_i} \cdot \Phi^{\delta}_{[n]\setminus V}(y).$$

Take $m'(y) = \beta \prod_{i \in V}(y - \alpha_i)^{\delta - e_i}$ and observe that degree of $m'(y)$ is $\delta \cdot |V| - d$. By definition, the polynomial P consists of a sum of such monomial where $V = \mathrm{var}(P)$. Therefore, the first claim follows by a linearity argument. The second claim follows by observing that d is an upper bound on the individual degrees of all variables x_i in P, so we can set $\delta = d$. ◀

4 Main Result

In this section we prove our main result Theorem 1. We begin by showing that $P(G_{n,1})$ hits sums of univariate polynomials. This proof is available in [52] but we reproduce it here for completeness.

▶ **Lemma 19.** *Let $P \in \mathbb{F}[x_1, \ldots, x_n]$ be a non-constant polynomial of the form $P = \sum_{i=1}^{n} T_i(x_i)$. Then $P(G_{n,1}(y,z))$ is non-constant.*

Proof. Pick x_i such that $T_i(x_i)$ is non-constant. Observe that: $P(G_{n,1})|_{y=\alpha_i} = T_i(z) + \sum_{j \neq i} T_j(0)$. This is a non-constant polynomial and so $P(G_{n,1})$ is non-constant as well. ◀

We now move to the proof our main result. We want to show that $G_{n,1}$ is a generator for the set of PROPs. The idea is to proceed by induction using Lemma 13. Recall that for any $P \in \mathbb{F}[x_1, \ldots, x_n]$ and $i \in \mathbb{N}$, $H_i[P]$ denotes the homogeneous part of degree i of P. Consequently, we can write $P = \sum_{i=0}^{d} H_i[P]$.

▶ **Theorem 20.** *Let $P \in \mathbb{F}[x_1, \ldots, x_n]$ be a non-constant PROP. Then $P(G_{n,1})$ is non-constant.*

Proof. Let d denote the total degree of P. We induct on $m = |\mathrm{var}(P)|$. The base case where $m = 1$ follows from Lemma 19. Now, suppose that $m \geq 2$. By the PROP Structural Lemma (Lemma 13) we have two cases.

1. $P = P_1 \cdot P_2 + c$. Note that $|\mathrm{var}(P_1)|, |\mathrm{var}(P_2)| \leq m - 1$, so by the inductive hypothesis $P_1(G_{n,1})$ and $P_2(G_{n,1})$ are non-constant polynomials and hence their product is non-constant as well. Adding a constant does not affect this.
2. $P = P_1 + P_2$. For $j = 1, 2$: we can write $P_j = \sum_{i=0}^{d} P_{i,j}$ where $P_{i,j} = H_i[P_j]$. By Lemma 18, for each $0 \leq i \leq d$ and $j = 1, 2$ there exists a polynomial $P'_{i,j}(y)$ of degree at most $i \cdot (|\mathrm{var}(P_{i,j})| - 1)$ such that

$$P_j(G_{n,1}(y,z)) = \sum_{i=0}^{d} P_{i,j}(G_{n,1}(y,z)) = \sum_{i=0}^{d} z^i \cdot P'_{i,j}(y) \cdot \Phi^i_{[n]\setminus \mathrm{var}(P_{i,j})}(y)$$

and hence

$$P(G_{n,1}(y,z)) = \sum_{i=0}^{d} z^i \cdot \left(P'_{i,1}(y) \cdot \Phi^i_{[n]\setminus \mathrm{var}(P_{i,1})}(y) + P'_{i,2}(y) \cdot \Phi^i_{[n]\setminus \mathrm{var}(P_{i,2})}(y) \right). \tag{1}$$

As before, by the inductive hypothesis $P_1(G_{n,1})$ and $P_2(G_{n,1})$ are non-constant polynomials. Therefore, there exist $1 \leq k \leq d$ such that

$$z^k \cdot P'_{k,1}(y) \cdot \Phi^k_{[n]\setminus \mathrm{var}(P_{k,1})}(y) \not\equiv 0$$

and in particular $P'_{k,1}(y) \not\equiv 0$. Let us denote $V_j = \mathrm{var}(P_{k,j})$ and $W = [n] \setminus (V_1 \cup V_2)$. Consider the expression that corresponds to the z^k term in Equation 1:

$$P'_{k,1}(y) \cdot \Phi^k_{[n]\setminus \mathrm{var}(P_{k,1})}(y) + P'_{k,2}(y) \cdot \Phi^k_{[n]\setminus \mathrm{var}(P_{k,2})}(y) \tag{2}$$

As $P_{k,1}$ and $P_{k,2}$ are variable-disjoint, Equation 2 can be rewritten as:

$$P'_{k,1}(y) \cdot \Phi^k_{V_2 \cup W}(y) + P'_{k,2}(y) \cdot \Phi^k_{V_1 \cup W}(y) =$$
$$\Phi^k_W(y) \cdot \left(P'_{k,1}(y) \cdot \Phi^k_{V_2}(y) + P'_{k,2}(y) \cdot \Phi^k_{V_1}(y) \right)$$

The last equality follows from the properties of Φ (see Definition 17).
We claim that the obtained expression is non-constant. To this end, it sufficient to show that $P'_{k,1}(y) \cdot \Phi^k_{V_2}(y) + P'_{k,2}(y) \cdot \Phi^k_{V_1}(y) \not\equiv 0$. Assume the contrary. We obtain that

$P'_{k,1}(y) \cdot \Phi^k_{V_2}(y) = -P'_{k,2}(y) \cdot \Phi^k_{V_1}(y)$. As V_1 and V_2 are disjoint sets, $\Phi_{V_1}(y)$ and $\Phi_{V_2}(y)$ have no common roots. Therefore, it must be the case that $\Phi^k_{V_1}$ divides $P'_{k,1}$. As $P'_{k,1} \not\equiv 0$, we get that

$$\deg\left(P'_{k,1}\right) \geq \deg\left(\Phi^k_{V_1}\right) = k\,|V_1|$$

while by Lemma 18, $P'_{k,1}(y)$ is a polynomial of degree at most $k \cdot (|V_1| - 1)$. Consequently, the coefficient of z^k in $P(G_{n,1}(y,z))$ is non-constant and the claim follows. ◀

Theorem 1 follows by combining Theorem 20 with Lemma 16.

5 Applications

In this section we show two application for our main result, proving Theorems 2 and 3. The first application is testing whether several PROPs sum up to the zero polynomial. To this end, we require the following result which shows that a generator for the class of PROPs can be extended to yield a generator for the class of sums of PROPs.

▶ **Lemma 21** ([52]). *Let $\mathcal{G}_n$ be a generator for PROPs on n variables. Then for any $k \in \mathbb{N}$, $\mathcal{G}_n + G_{n,3k}$ is a generator for sums of k PROPs on n variables.*

▶ Remark. As both $\mathcal{G}_n$ and $G_{n,3k}$ represent polynomial maps with the same output length, the sum $\mathcal{G} + G_{n,3k}$ should be interpreted as component-wise sum, where we implicitly assume the variables of $\mathcal{G}_n$ and $G_{n,3k}$ have been relabelled so as to be *disjoint*.

The next corollary follows by combining Lemma 21 with Theorem 20 and the properties of $G_{n,t}$ (see Definition 14).

▶ **Corollary 22.** *For any $k, n \in \mathbb{N}$, the map $G_{n,3k+1}$ is a generator for sums of k PROPs on n variables.*

Theorem 2 follows by applying Lemma 16. Observe that if $\mathcal{H}$ is hitting set for a sum of two PROPs then $\mathcal{H}$ is an *interpolating* set for a single PROP. That is, the values of a single PROP P on $\mathcal{H}$ contain enough information to uniquely identify P. Indeed, a consequence of Theorem 2 is an interpolating set of polynomial size for PROPs. However in general, $\mathcal{H}$ does not provide us with an efficient algorithm to reconstruct a corresponding PROF. In [51] it was shown how to use an interpolating set to devise a reconstruction algorithm with a polynomial overhead.

▶ **Lemma 23** ([51]). *Let $n, d \in \mathbb{N}$. There exists a deterministic algorithm that given a hitting set $\mathcal{H}_{n,d}$ for PROPs on n variable and degree at most d, and black-box (oracle) access to a PROP P as above, outputs a PROF Φ that computes P, in time polynomial in n, d and $|\mathcal{H}_{n,d}|$.*

Combining the Lemma with Theorem 1 results in Theorem 3.

6 Conclusions & Open Questions

In this paper we present the first polynomial-time black-box identity testing and reconstruction algorithms for read-once formulas, which form a subclass of multilinear formulas. In [7], quasi-polynomial-time and polynomial-time PIT algorithms were given for multilinear read-k formulas in the black-box and the white-box settings, respectively for constant values of k. At a high-level, both algorithms go by alternating the following two steps:

- Step 1: Reduce PIT of a read-$(k+1)$ formula to PIT of sum of two read-k formulas.
- Step 2: Reduce PIT of sum of two read-k formulas to PIT of a (single) read-k formula.

While Step 2 introduces an overhead of (roughly) $n^{k^{\mathcal{O}(k)}}$ in both settings, the gap in the final complexity results from the overhead introduced by Step 1. Indeed, in the whitebox setting, the overhead is $\mathrm{poly}(n,k)$ while in black-box setting the overhead is $n^{\mathcal{O}(\log n)}$. Moreover, for $k=0$ the the analysis of Step 2 can be seen as a different analysis of the black-box PIT algorithm for ROFs of [52], resulting in roughly the same run time. We hope that the ideas presented in this paper could be extended further to improve the analysis of the black-box PIT algorithms of [7], and, perhaps lead to new PIT algorithms.

Some open questions: can one obtain a polynomial-time black-box PIT algorithm for multilinear read-k formula with a constant k? What about $k=2$? I.e. multilinear read-twice formulas. Even more specifically, can one show a black-box reduction from a PIT instance of a multilinear read-twice formula to polynomially-many PIT instances of sums of constantly-many read-once formulas, introducing only a polynomial overhead?

Acknowledgments. The authors would like to thank the anonymous referees for useful comments.

References

1 M. Agrawal. Proving lower bounds via pseudo-random generators. In *Proceedings of the 25th FSTTCS*, volume 3821 of *LNCS*, pages 92–105, 2005.

2 M. Agrawal, N. Kayal, and N. Saxena. Primes is in P. *Annals of Mathematics*, 160(2):781–793, 2004.

3 M. Agrawal, C. Saha, R. Saptharishi, and N. Saxena. Jacobian hits circuits: Hitting-sets, lower bounds for depth-d occur-k formulas & depth-3 transcendence degree-k circuits. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC)*, pages 599–614, 2012.

4 M. Agrawal, C. Saha, and N. Saxena. Quasi-polynomial hitting-set for set-depth- formulas. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing (STOC)*, pages 321–330, 2013.

5 N. Alon. Combinatorial nullstellensatz. *Combinatorics, Probability and Computing*, 8:7–29, 1999.

6 M. Anderson, M. A. Forbes, R. Saptharishi, A. Shpilka, and B. L. Volk. Identity Testing and Lower Bounds for Read-k Oblivious Algebraic Branching Programs. In *31st Conference on Computational Complexity (CCC 2016)*, volume 50 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 30:1–30:25. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.CCC.2016.30.

7 M. Anderson, D. van Melkebeek, and I. Volkovich. Derandomizing polynomial identity testing for multilinear constant-read formulae. *Computational Complexity*, 24(4):695–776, 2015.

8 D. Angluin, L. Hellerstein, and M. Karpinski. Learning read-once formulas with queries. *J. ACM*, 40(1):185–210, 1993.

9 S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and the hardness of approximation problems. *JACM*, 45(3):501–555, 1998.

10 S. Arora and S. Safra. Probabilistic checking of proofs: A new characterization of NP. *JACM*, 45(1):70–122, 1998.

11 V. Arvind and P. Mukhopadhyay. The monomial ideal membership problem and polynomial identity testing. *Information and Computation*, 208(4):351–363, 2010.

12 M. Beecken, J. Mittmann, and N. Saxena. Algebraic independence and blackbox identity testing. In *Automata, Languages and Programming, 38th International Colloquium (ICALP)*, pages 137–148, 2011.

13 M. Ben-Or and P. Tiwari. A deterministic algorithm for sparse multivariate polynominal interpolation. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing (STOC)*, pages 301–309, 1988.

14 D. Bshouty and N. H. Bshouty. On interpolating arithmetic read-once formulas with exponentiation. *JCSS*, 56(1):112–124, 1998.

15 N. H. Bshouty and R. Cleve. Interpolating arithmetic read-once formulas in parallel. *SIAM J. on Computing*, 27(2):401–413, 1998.

16 N. H. Bshouty, T. R. Hancock, and L. Hellerstein. Learning arithmetic read-once formulas. *SIAM J. on Computing*, 24(4):706–735, 1995.

17 N. H. Bshouty, T. R. Hancock, and L. Hellerstein. Learning boolean read-once formulas with arbitrary symmetric and constant fan-in gates. *JCSS*, 50:521–542, 1995.

18 R. A. DeMillo and R. J. Lipton. A probabilistic remark on algebraic program testing. *Inf. Process. Lett.*, 7(4):193–195, 1978.

19 Z. Dvir and A. Shpilka. Locally decodable codes with 2 queries and polynomial identity testing for depth 3 circuits. *SIAM J. on Computing*, 36(5):1404–1434, 2006.

20 S. A. Fenner, R. Gurjar, and T. Thierauf. Bipartite perfect matching is in quasi-nc. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC*, pages 754–763, 2016. doi:10.1145/2897518.2897564.

21 M. Forbes, R. Saptharishi, and A. Shpilka. Pseudorandomness for multilinear read-once algebraic branching programs, in any order. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC)*, pages 867–875, 2014. Full version at https://eccc.weizmann.ac.il/report/2013/132/. doi:10.1145/2591796.2591816.

22 M. Forbes and A. Shpilka. Quasipolynomial-time identity testing of non-commutative and read-once oblivious algebraic branching programs. *Electronic Colloquium on Computational Complexity (ECCC)*, 19:115, 2012.

23 M. Forbes and A. Shpilka. Explicit noether normalization for simultaneous conjugation via polynomial identity testing. In *APPROX-RANDOM*, pages 527–542, 2013.

24 M. Forbes and A. Shpilka. Quasipolynomial-time identity testing of non-commutative and read-once oblivious algebraic branching programs. In *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 243–252, 2013. Full version at https://eccc.weizmann.ac.il/report/2012/115/. doi:10.1109/FOCS.2013.34.

25 M. A. Forbes, A. Shpilka, I. Tzameret, and A. Wigderson. Proof complexity lower bounds from algebraic circuit complexity. *CoRR*, abs/1606.05050, 2016. URL: http://arxiv.org/abs/1606.05050.

26 A. Gupta. Algebraic geometric techniques for depth-4 PIT & sylvester-gallai conjectures for varieties. *Electronic Colloquium on Computational Complexity (ECCC)*, 21:130, 2014. URL: https://eccc.weizmann.ac.il/report/2014/130/.

27 R. Gurjar, A. Korwar, and N. Saxena. Identity Testing for Constant-Width, and Commutative, Read-Once Oblivious ABPs. In *31st Conference on Computational Complexity (CCC 2016)*, volume 50 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 29:1–29:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.CCC.2016.29.

28 R. Gurjar, A. Korwar, N. Saxena, and T. Thierauf. Deterministic Identity Testing for Sum of Read-once Oblivious Arithmetic Branching Programs. In *30th Conference on Computational Complexity (CCC 2015)*, volume 33 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 323–346. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2015. doi:10.4230/LIPIcs.CCC.2015.323.

29 T. R. Hancock and L. Hellerstein. Learning read-once formulas over fields and extended bases. In *Proceedings of the 4th Annual Workshop on Computational Learning Theory (COLT)*, pages 326–336, 1991.

30 J. Heintz and C. P. Schnorr. Testing polynomials which are easy to compute (extended abstract). In *Proceedings of the 12th Annual ACM Symposium on Theory of Computing (STOC)*, pages 262–272, 1980.

31 M. Karchmer, N. Linial, I. Newman, M. E. Saks, and A. Wigderson. Combinatorial characterization of read-once formulae. *Discrete Mathematics*, 114(1-3):275–282, 1993.

32 Z. S. Karnin, P. Mukhopadhyay, A. Shpilka, and I. Volkovich. Deterministic identity testing of depth 4 multilinear circuits with bounded top fan-in. *SIAM J. on Computing*, 42(6):2114–2131, 2013.

33 Z. S. Karnin and A. Shpilka. Black box polynomial identity testing of generalized depth-3 arithmetic circuits with bounded top fan-in. *Combinatorica*, 31(3):333–364, 2011. doi:10.1007/s00493-011-2537-3.

34 N. Kayal and S. Saraf. Blackbox polynomial identity testing for depth 3 circuits. In *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 198–207, 2009. Full version at https://eccc.weizmann.ac.il/report/2009/032/.

35 N. Kayal and N. Saxena. Polynomial identity testing for depth 3 circuits. *Computational Complexity*, 16(2):115–138, 2007.

36 A. Klivans and D. Spielman. Randomness efficient identity testing of multivariate polynomials. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 216–223, 2001.

37 M. Kumar and S. Saraf. Arithmetic Circuits with Locally Low Algebraic Rank. In *31st Conference on Computational Complexity (CCC 2016)*, volume 50 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 34:1–34:27. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.CCC.2016.34.

38 M. Kumar and S. Saraf. Sums of Products of Polynomials in Few Variables: Lower Bounds and Polynomial Identity Testing. In *31st Conference on Computational Complexity, CCC*, volume 50 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 35:1–35:29. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.CCC.2016.35.

39 R. J. Lipton and N. K. Vishnoi. Deterministic identity testing for multivariate polynomials. In *Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 756–760, 2003.

40 L. Lovasz. On determinants, matchings, and random algorithms. In L. Budach, editor, *Fundamentals of Computing Theory*. Akademia-Verlag, 1979.

41 C. Lund, L. Fortnow, H. Karloff, and N. Nisan. Algebraic methods for interactive proof systems. *JACM*, 39(4):859–868, 1992.

42 P. Mukhopadhyay. Depth-4 identity testing and noether's normalization lemma. *Electronic Colloquium on Computational Complexity (ECCC)*, 2015.

43 K. Mulmuley, U. Vazirani, and V. Vazirani. Matching is as easy as matrix inversion. *Combinatorica*, 7(1):105–113, 1987.

44 N. Saxena. Diagonal circuit identity testing and lower bounds. In *Automata, Languages and Programming, 35th International Colloquium*, pages 60–71, 2008. Full version at https://eccc.weizmann.ac.il/eccc-reports/2007/TR07-124/.

45 N. Saxena and C. Seshadhri. From Sylvester-Gallai Configurations to Rank Bounds: Improved Black-Box Identity Test for Deph-3 Circuits. In *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 21–30, 2010.

46 N. Saxena and C. Seshadhri. An almost optimal rank bound for depth-3 identities. *SIAM J. Comput.*, 40(1):200–224, 2011.

47 J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, 27(4):701–717, 1980.

48 A. Shamir. IP=PSPACE. In *Proceedings of the Thirty First Annual Symposium on Foundations of Computer Science*, pages 11–15, 1990.

49 A. Shpilka and I. Volkovich. Read-once polynomial identity testing. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC)*, pages 507–516, 2008. doi:10.1145/1374376.1374448.

50 A. Shpilka and I. Volkovich. Improved polynomial identity testing for read-once formulas. In *APPROX-RANDOM*, pages 700–713, 2009. Full version at https://eccc.weizmann.ac.il/report/2010/011/.

51 A. Shpilka and I. Volkovich. On reconstruction and testing of read-once formulas. *Theory of Computing*, 10:465–514, 2014.

52 A. Shpilka and I. Volkovich. Read-once polynomial identity testing. *Computational Complexity*, 24(3):477–532, 2015.

53 A. Shpilka and A. Yehudayoff. Arithmetic circuits: A survey of recent results and open questions. *Foundations and Trends in Theoretical Computer Science*, 5(3-4):207–388, 2010.

54 I. Volkovich. Characterizing arithmetic read-once formulae. *ACM Transactions on Computation Theory (ToCT)*, 8(1):2, 2016. doi:10.1145/2858783.

55 R. Zippel. Probabilistic algorithms for sparse polynomials. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation*, pages 216–226, 1979.

Greedy Strikes Again: A Deterministic PTAS for Commutative Rank of Matrix Spaces

Markus Bläser[1], Gorav Jindal[2], and Anurag Pandey[3]

1 **Department of Computer Science, Saarland University, Saarland Informatics Campus, Saarbrücken, Germany**
mblaeser@cs.uni-saarland.de
2 **Max-Planck-Institute for Informatics & Saarbrücken Graduate School of Computer Science, Saarland Informatics Campus, Saarbrücken, Germany**
gjindal@mpi-inf.mpg.de
3 **Max-Planck-Institute for Informatics & Saarbrücken Graduate School of Computer Science, Saarland Informatics Campus, Saarbrücken, Germany**
apandey@mpi-inf.mpg.de

Abstract

We consider the problem of commutative rank computation of a given matrix space, $\mathcal{B} \subseteq \mathbb{F}^{n\times n}$. The problem is fundamental, as it generalizes several computational problems from algebra and combinatorics. For instance, checking if the commutative rank of the space is n, subsumes problems such as testing perfect matching in graphs and identity testing of algebraic branching programs. An efficient deterministic computation of the commutative rank is a major open problem, although there is a simple and efficient randomized algorithm for it. Recently, there has been a series of results on computing the non-commutative rank of matrix spaces in deterministic polynomial time. Since the non-commutative rank of any matrix space is at most twice the commutative rank, one immediately gets a deterministic $\frac{1}{2}$-approximation algorithm for the computation of the commutative rank. This leads to a natural question of whether this approximation ratio can be improved. In this paper, we answer this question affirmatively.

We present a deterministic Polynomial-time approximation scheme (PTAS) for computing the commutative rank of a given matrix space. More specifically, given a matrix space $\mathcal{B} \subseteq \mathbb{F}^{n\times n}$ and a rational number $\epsilon > 0$, we give an algorithm, that runs in time $O(n^{4+\frac{3}{\epsilon}})$ and computes a matrix $A \in \mathcal{B}$ such that the rank of A is at least $(1-\epsilon)$ times the commutative rank of $\mathcal{B}$. The algorithm is the natural greedy algorithm. It always takes the first set of k matrices that will increase the rank of the matrix constructed so far until it does not find any improvement, where the size of the set k depends on ϵ.

1998 ACM Subject Classification G.1.3 Numerical Linear Algebra

Keywords and phrases Commutative Rank, Matrix Spaces, PTAS, Wong sequences

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.33

1 Introduction

In this paper, we consider the problem of computing the maximum rank of any matrix which lies in the linear span of m given input $n \times n$ matrices $B_1, B_2, \ldots, B_m$ over some underlying field $\mathbb{F}$. This maximum rank is also called the *commutative rank* of the matrix space $\mathcal{B} = \langle B_1, B_2, \ldots, B_m\rangle$. This problem was introduced by Edmonds in [3]. Any matrix spanned by $B_1, B_2, \ldots, B_m$ can be written as the homomorphic image of $B = \sum_{i=1}^{m} x_i B_i$ under the substitution homomorphism, where we think of the x_i as indeterminates. It is not hard to see that the commutative rank of the $\mathcal{B}$ is same as the rank of B over the field of

32nd Conference on Computational Complexity (CCC 2017).
Editor: Ryan O'Donnell; Article No. 33; pp. 33:1–33:16


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

rational functions $\mathbb{F}(x_1, x_2, \ldots, x_m)$, provided that $\mathbb{F}$ is large enough. For this reason, this problem is also called the *symbolic matrix rank* sometimes. Since the rank of B is the size of the largest nonzero minor in B and any minor of B is a polynomial of degree at most n in the variables $x_1, x_2, \ldots, x_m$, by using the Schwartz-Zippel lemma [22, 18], one immediately gets a randomized algorithm for computing the commutative rank of $\mathcal{B}$ if the size of the field $\mathbb{F}$ is large enough. The maximum matching problem in bipartite and general graphs is a special case of the commutative rank problem, as shown in [13]. Even the linear matroid parity problem is special case of the commutative rank problem [17].

Valiant [20] showed that a formula of size s can be written as a projection of the determinant of an $(s+2) \times (s+2)$ matrix having linear polynomials as entries. This shows that checking if a given matrix space is full rank is as hard as polynomial identity testing of formulas. In fact, it is even known that algebraic branching programs are computationally equivalent to the polynomials computed by determinants of a polynomial sized matrix, see [21, 19, 16]. So the problem of deciding whether a given matrix space is full rank is as hard as the polynomial identity testing of arithmetic branching programs. Algebraic branching programs are conjectured to be a stronger model for computing polynomials than formulas.

We remark that if the underlying field $\mathbb{F}$ is not large enough, then this problem is hard. Buss et al. proved that the problem is NP-complete in [1], when the field $\mathbb{F}$ is of constant size.

1.1 Previous work

Since the general case of computing the commutative rank is as hard as identity testing for polynomials given as algebraic branching programs, several special cases of matrix spaces have been considered. There has been a lot of study in the case when all the matrices B_i are of rank 1 [14, 9, 10]. Deterministic polynomial time algorithms were shown for this case in [9, 10]. The case when the matrices B_i are skew-symmetric of rank 2 is also of special interest as it was shown in [13] that the linear matroid parity problem is a special case of computing the commutative rank when B_i are skew-symmetric of rank 2. Many deterministic polynomial time algorithms have been demonstrated for this case, see [12, 6, 15].

Analogous to the notion of commutative rank of a matrix space, there is also a notion of non-commutative rank (see the next section for a precise definition). The matrix spaces for which commutative rank and non-commutative rank are equal are called *compression spaces* [4]. A deterministic polynomial time algorithm for checking if a compression space is of full rank (over the field $\mathbb{Q}$) was discovered by Gurvits in [8]. The algorithm of [8] was analysed more carefully in [7] to demonstrate that the algorithm described in [8] actually is a deterministic polynomial time algorithm to check if a given matrix space has full non-commutative rank. This algorithm works over $\mathbb{Q}$ only. Ivanyos et al. [11] extended this results to arbitrary fields, using a totally different algorithm. It was shown in [5] that non-commutative of any matrix space is at most twice the commutative rank. So the algorithms in [7, 11] are deterministic polynomial time algorithms which compute a $\frac{1}{2}$-approximation to the commutative rank. Approximating the commutative rank of a matrix space can be seen as a relaxation of the polynomial identity testing problem. Improving on the $\frac{1}{2}$-approximation was formulated as an open problem in [7].

1.2 Our results

We here improve on this approximation performance. We give a deterministic polynomial time approximation scheme (PTAS) for approximating the commutative rank. That is, given a basis $B_1, \ldots, B_m$ of our matrix space $\mathcal{B}$ of $n \times n$-matrices and some rational number

$\epsilon > 0$, our algorithm outputs a matrix $A \in \mathcal{B}$ whose rank is at least $(1-\epsilon) \cdot r$, where $r = \max\{\operatorname{rank}(B) \mid B \in \mathcal{B}\}$ provided that the size of the underlying field is larger than n. Our algorithm performs $O(n^{4+\frac{3}{\epsilon}})$ many arithmetic operations, the size of each operand is linear in the sizes of the entries of the matrices $B_1, \ldots, B_m$. So for fixed ϵ, the running time is polynomial in the input size.

Our algorithm is the natural greedy algorithm: Assume we have constructed a matrix A so far. Then the algorithm tries all subsets of $B_1, \ldots, B_m$ of size k, where k depends on ϵ, and tests whether we can increase the rank of A by adding an appropriate linear combination of $B_{i_1}, \ldots, B_{i_k}$. The main difficulty is to prove that when this algorithm stops, A is an $(1-\epsilon)$-approximation. The analysis uses so-called Wong sequences.

For polynomial identity testing, one has to test whether a matrix has full rank or rank $\leq n-1$. Therefore, our PTAS does not seem to help getting a polynomial time algorithm for polynomial identity testing.

1.3 Organization of the paper

Section 2 describes the basic setup of the problem and relevant definitions and techniques. It describes the basic notations, definitions and related lemmas and theorems known. In Section 3, we first present a greedy algorithm which computes a $\frac{1}{2}$-approximation of the commutative rank in deterministic polynomial time. It describes the basic ideas of our algorithm but is much easier to analyse. This motivates our final algorithm which can compute arbitrary approximations to the commutative rank in deterministic polynomial time. To extend this $\frac{1}{2}$-approximation to arbitrary approximation, we introduce the notion of Wong sequences and Wong index in Section 4. Section 5 studies the relation between commutative rank and Wong index. In this section, we prove that the higher the Wong index is of a given matrix, the closer its rank is to the commutative rank of the given matrix space. This allows us to extend Algorithm 1 to arbitrary approximation by considering larger subsets. The algorithm for arbitrary approximation of the commutative rank and its proof of correctness and desired running time are given in Section 6. We conclude by giving some tight examples in Section 7.

2 Preliminaries

Here, we introduce the basic definitions and notations which are needed to fully describe our algorithm.

1. If V and W are vector spaces, then we use notation $V \leq W$ to denote that V is a subspace of W.
2. We use $\mathbb{F}^{n\times n}$ to denote the set of all $n \times n$ matrices over a field $\mathbb{F}$.
3. $\operatorname{Im}(A)$ is used to denote the image of a matrix $A \in \mathbb{F}^{n\times n}$.
4. $\operatorname{Ker}(A)$ is used to denote the kernel of a linear map $A \in \mathbb{F}^{n\times n}$.
5. $\dim(V)$ is used to denote the dimension of a vector space V.
6. For any subset S of a vector space U, $\langle S\rangle$ denotes the linear span of S.
7. For $A \in \mathbb{F}^{n\times n}$ and a vector space $U \leq \mathbb{F}^n$, the image of U under A is $A(U) = AU = \{A(u) \mid u \in U\}$.
8. The preimage of $W \leq \mathbb{F}^n$ under A is defined as $A^{-1}(W) = \{v \in V \mid A(v) \in W\}$.
9. The set $\{0, 1, 2, \ldots, n\}$ of non-negative integers between 0 and n is denoted by $[n]$.
10. We use the notation I_r to denote the $r \times r$ identity matrix.
11. Throughout the paper, we would assume that the size of the underlying field is more than n, the size of the input matrices, i.e., $|\mathbb{F}| > n$.

Below are some of the basic definitions which we shall need.

▶ **Definition 1** (Matrix space)**.** A vector space $\mathcal{B} \leq \mathbb{F}^{n\times n}$ is called a *matrix space*.

We would usually deal with matrix spaces whose generating set is given as the input. More precisely, we would be given a matrix space $\mathcal{B} = \langle B_1, B_2, \ldots, B_m\rangle \leq \mathbb{F}^{n\times n}$, where we get the matrices $B_1, B_2, \ldots, B_m$ as the input. Note that without loss of generality, one can assume that $m \leq n^2$.

▶ **Definition 2** (Commutative rank)**.** The maximum rank of any matrix in a matrix space $\mathcal{B}$ is called the *commutative rank* of $\mathcal{B}$. We use notation $\text{rank}(\mathcal{B})$ to denote this quantity.

We shall use the same notation $\text{rank}(A)$ for denoting the usual rank of any matrix. Note that the rank of a matrix A is same as the commutative rank of the matrix space generated by A, that is, $\text{rank}(A) = \text{rank}(\langle A\rangle)$.

▶ **Definition 3** (Product of a matrix space and a vector space)**.** The image of a vector space U under a matrix space $\mathcal{A}$ is the span of the images of U under every $A \in \mathcal{A}$, that is $\mathcal{A}(U) := \mathcal{A}U := \langle \bigcup_{A\in\mathcal{A}} A(U)\rangle$. We also call this image $\mathcal{A}U$ to be the product of the matrix space $\mathcal{A}$ and the vector space U.

▶ **Definition 4** (c-shrunk subspace)**.** A vector space $V \leq \mathbb{F}^n$ is a *c-shrunk subspace* of a matrix space $\mathcal{B}$, if $\text{rank}(\mathcal{B}V) \leq \dim(V) - c$.

▶ **Definition 5** (Non-commutative rank)**.** Given a matrix space $\mathcal{B} \leq \mathbb{F}^{n\times n}$, let r be the maximum non-negative integer such that there exists a r-shrunk subspace of the matrix space $\mathcal{B}$. Then $n - r$ is called the *non-commutative rank* of $\mathcal{B}$. We use the notation $\text{nc-rank}(\mathcal{B})$ to denote this quantity.

From the definition above, it is not clear why we call this quantity non-commutative rank. It can be shown that the quantity above equals the rank of the corresponding symbolic matrix when the variables $x_1, \ldots, x_m$ do not commute. For more natural and equivalent definitions as well as more background on non-commutative rank, we refer the reader to [7, 5].

▶ **Lemma 6.** *For all matrix spaces $\mathcal{B} \leq \mathbb{F}^{n\times n}$,* $\text{rank}(\mathcal{B}) \leq \text{nc-rank}(\mathcal{B})$.

Above lemma states that the non-commutative rank is at least as large as the commutative rank. But how large it can be compared to the commutative rank? Following theorem states that it is always less than twice the commutative rank.

▶ **Theorem 7** ([5], [2])**.** *For all matrix spaces $\mathcal{B} \leq \mathbb{F}^{n\times n}$, we have* $\frac{\text{nc-rank}(\mathcal{B})}{\text{rank}(\mathcal{B})} < 2$.

Derksen and Makam also gave a family of examples where the ratio of non-commutative rank and commutative rank reaches arbitrarily closed to 2, hence showing that the bound above is sharp (see [2], Theorem 1.15).

3 $\frac{1}{2}$-approximation algorithm for the commutative rank

Here we present a simple greedy algorithm which also achieves an $\frac{1}{2}$-approximation for the commutative rank. This algorithm looks for the first matrix that increases the rank of the current matrix and stops if it does not find such a matrix.

Input : A matrix space $\mathcal{B} = \langle B_1, B_2, , \ldots, B_m \rangle \leq \mathbb{F}^{n \times n}$, input is a list of matrices $B_1, B_2, \ldots, B_m$.
Output : A matrix $A \in \mathcal{B}$ such that $\operatorname{rank}(A) \geq \frac{1}{2} \cdot \operatorname{rank}(\mathcal{B})$
Initialize $A = 0 \in \mathbb{F}^{n \times n}$ to the zero matrix.
while *Rank is increasing* **do**
  for *each* $1 \leq i \leq m$ **do**
    Check if there exists a $\lambda \in \mathbb{F}$ such that $\operatorname{rank}(A + \lambda B_i) > \operatorname{rank}(A)$. **if** $\operatorname{rank}(A + \lambda B_i) > \operatorname{rank}(A)$ **then**
      Update $A = A + \lambda B_i$.
return A.

Algorithm 1 Greedy algorithm for $\frac{1}{2}$-approximating commutative rank.

We shall proof the following lemma in appendix.

▶ **Lemma 8.** *Algorithm 1 runs in polynomial time and returns a matrix $A \in \mathcal{B}$ such that* $\operatorname{rank}(A) \geq \frac{1}{2} \cdot \operatorname{rank}(\mathcal{B})$.

4 Wong sequences and Wong index

In this section, we introduce the notion of Wong sequences which is crucially used in our proofs. For a more comprehensive exposition, we refer reader to [9].

▶ **Definition 9** (Second Wong Sequence)**.** Let $\mathcal{B} \leq \mathbb{F}^{n \times n}$ be a matrix space and $A \in \mathcal{B}$. The sequence of sub-spaces $(W_i)_{i \in [n]}$ of W is called the *second Wong sequence* of $(A, \mathcal{B})$, where $W_0 = \{0\}$, and $W_{i+1} = \mathcal{B}A^{-1}(W_i)$.

In [9], first Wong sequences are also introduced. But for our purpose, just the notion of *second Wong sequence* is enough. It is easy to see that $W_0 \leq W_1 \leq W_2 \leq \ldots \leq W_n$, see [9].

Next, we introduce the notion of pseudo-inverses. They are helpful in computing the Wong sequences. We remark that we would need the notion of Wong sequence only for the analysis, our algorithm is completely oblivious to Wong sequences.

▶ **Definition 10** (Pseudo-Inverse)**.** A non-singular matrix $A' \in \mathbb{F}^{n \times n}$ is called a *pseudo-inverse* of a linear map $A \in \mathbb{F}^{n \times n}$ if the restriction of A' to $\operatorname{Im}(A)$ is the inverse of the restriction of A to a direct complement of $\operatorname{Ker}(A)$.

Unlike the usual inverse of a non-singular matrix, a pseudo-inverse of a matrix is not necessarily unique. But it always exists and if A is non-singular, then it is unique and coincides with the usual inverse.

The following lemma demonstrates the role of pseudo-inverses in computing Wong sequences. This lemma and its proof are implicit in the proof of Lemma 10 in [9]. We prove it in the appendix for completeness. The lemma essentially states that we can replace the preimage computation in the Wong sequence by multiplication with a pseudo-inverse.

▶ **Lemma 11.** *Let $\mathcal{B} \leq \mathbb{F}^{n \times n}$ be a matrix space, $A \in \mathcal{B}$, A' be a pseudo-inverse of A and $(W_i)_{i \in [n]}$ be the second Wong sequence of $(A, \mathcal{B})$. Then for all $1 \leq i \leq n$, we have $W_i = (\mathcal{B}A')^i(\operatorname{Ker}(AA'))$ as long as $W_{i-1} \subseteq \operatorname{Im} A$.*

Given a matrix space $\mathcal{B}$ and a matrix $A \in \mathcal{B}$, how can one check that A is of maximum rank in $\mathcal{B}$, i.e, $\operatorname{rank}(A) = \operatorname{rank}(\mathcal{B})$? The following lemma in [9] gives a sufficient condition for A to be of maximum rank in $\mathcal{B}$.

▶ **Lemma 12** (Lemma 10 in [9]). *Assume that $|\mathbb{F}| > n$. Let $A \in \mathcal{B} \leq \mathbb{F}^{n\times n}$, and let A' be a pseudo-inverse of A. If we have that for all $i \in [n]$,*

$$W_i = (\mathcal{B}A')^i(\mathrm{Ker}(AA')) \subseteq \mathrm{Im}(A), \tag{4.1}$$

then A is of maximum rank in $\mathcal{B}$.

Thus, the above lemma shows that if A is not of maximum rank in $\mathcal{B}$, then we have $W_i \nsubseteq \mathrm{Im}(A)$ for some $i \in [n]$. For our purposes, we need to quantify when exactly this happens. Therefore we define:

▶ **Definition 13** (Wong Index). Let $\mathcal{B} \leq \mathbb{F}^{n\times n}$ be a matrix space, $A \in \mathcal{B}$ and $(W_i)_{i\in[n]}$ be the second Wong sequence of $(A, \mathcal{B})$. Let $k \in [n]$ be the maximum integer such that $W_k \subseteq \mathrm{Im}(A)$. Then k is called the *Wong index* of $(A, \mathcal{B})$. We shall denote it by $w(A, \mathcal{B})$.

Using the above definition, another way to state Lemma 12 is that if the Wong index $w(A, \mathcal{B})$ of $(A, \mathcal{B})$ is n, then A is of maximum rank in $\mathcal{B}$. But can one say more in this case? In next section, we explore this connection. We shall prove that the closer $w(A, \langle A, B\rangle)$ is to n, the closer the rank of A is to the commutative rank of $\langle A, B\rangle$.

The converse of Lemma 12 is not true in general. But the converse is true in the special case when $\mathcal{B}$ is spanned by just two matrices. Fortunately, for our algorithm we only require the converse to be true in this special case. The following fact from [9] formally states this idea.

▶ **Fact 14** (Restatement of Fact 11 in [9]). *Assume that $|\mathbb{F}| > n$ and let $A, B \in \mathbb{F}^{n\times n}$. If A is of maximum rank in $\langle A, B\rangle$ then the Wong index $w(A, \langle A, B\rangle)$ of $(A, \langle A, B\rangle)$ is n.*

We shall also need the following easy fact from linear algebra.

▶ **Fact 15.** *Let M be a matrix of the following form.*

$$M = \begin{array}{c} \overbrace{}^{r \text{ columns}} \\ \left.\begin{pmatrix} L & B \\ A & \underbrace{\mathbf{0}}_{n-r \text{ columns}} \end{pmatrix}\right\} \begin{array}{l} r \text{ rows} \\ n-r \text{ rows} \end{array} \end{array} \tag{4.2}$$

Also, let $\mathrm{rank}(A) = a$ and $\mathrm{rank}(B) = b$. Then $\mathrm{rank}(M) \leq r + \min\{a, b\}$.

In order to extend the simple greedy algorithm for rank increment described in Section 3 for arbitrary approximation of the commutative rank, we use the Wong index defined above. To achieve that, we need the relation between the commutative rank and Wong index, which we establish in the next section.

5 Relation between rank and Wong index

We prove that the natural greedy strategy works, essentially by showing that either of the following happens:

1. The Wong index of the matrix obtained by the greedy algorithm at a given step is high enough, in which case, we show that the matrix already has the desired rank. Lemma 19 formalizes this.
2. We can increase the rank by a greedy step. Lemma 20 formalizes this.

In the above spirit, we quantify the connection between the commutative rank and Wong index in this section, using a series of lemmas. First we need a lemma which demonstrates

that the second Wong sequence remains "almost" the same under invertible linear maps, which we prove in the appendix.

▶ **Lemma 16.** *Let $A \in \mathcal{B} \leq \mathbb{F}^{n\times n}$ and $(W_i)_{i\in[n]}$ be the second Wong sequence of $(A, \mathcal{B})$. If $P \in \mathbb{F}^{n\times n}$ and $Q \in \mathbb{F}^{n\times n}$ are invertible matrices, then the second Wong sequence of $(PAQ, P\mathcal{B}Q)$ is $(PW_i)_{i\in[n]}$. In particular, $w(A, \mathcal{B}) = w(PAQ, P\mathcal{B}Q)$.*

The following technical lemma relates Wong index with a sequence of vanishing matrix products.

▶ **Lemma 17.** *Let $A, B \in \mathbb{F}^{n\times n}$. Assume $A = \begin{bmatrix} I_r & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}$ and express the matrix B as*

$$B = \begin{array}{c} \\ r \text{ rows} \\ \\ n-r \text{ rows} \end{array} \overset{\overset{r \text{ columns}}{\overbrace{}}}{\left(\begin{array}{cc} B_{11} & B_{12} \\ B_{21} & \underbrace{B_{22}}_{n-r \text{ columns}} \end{array}\right)} \tag{5.1}$$

Let $\ell \leq n$ be the maximum integer such that first ℓ elements of the sequence of matrices

$$B_{22}, B_{21}B_{12}, B_{21}B_{11}B_{12}, \ldots, B_{21}B_{11}^i B_{12} \ldots. \tag{5.2}$$

are equal to the zero matrix. Then $\ell = w(A, \langle A, B\rangle)$.

Proof. Notice that I_n is a pseudo-inverse of A. Consider the second Wong sequence of $(A, \langle A, B\rangle)$. By Lemma 11, it equals $(\langle A, B\rangle A')^i(\mathrm{Ker}(AA'))$. Since $A' = I_n$, this sequence is $(\langle A, B\rangle)^i(\mathrm{Ker}(A))$. $\mathrm{Ker}(A) \leq \mathbb{F}^n$ contains exactly the vectors which have first r entries to be zero and $\mathrm{Im}(A)$ contains exactly the vectors which have last $n-r$ entries to be zero. Let $k = w(A, \langle A, B\rangle)$, we want to show that $k = \ell$.

First we show that $\ell \geq k$. For this, we need to show that $B_{22} = B_{21}B_{12} = B_{21}B_{11}B_{12} = \ldots = B_{21}B_{11}^{k-2}B_{12} = \mathbf{0}$. If $k = 0$ then we do not need to show anything. Otherwise $k > 0$. Consider the first entry W_1 of second Wong sequence of $(A, \langle A, B\rangle)$. By Lemma 11, we know that $W_1 = \langle A, B\rangle \mathrm{Ker}(A)$. As $\mathrm{Ker}(A) \leq \mathbb{F}^n$ contains exactly the vectors which have first r entries to be zero, if B_{22} was not zero then $B\,\mathrm{Ker}(A)$ would contain a vector with a non-zero entry in last $n-r$ coordinates. This would violate the assumption $W_1 \subseteq \mathrm{Im}(A)$. Thus $B_{22} = \mathbf{0}$. Now we use induction on length of the sequence $B_{22}, B_{21}B_{12}, B_{21}B_{11}B_{12}, \ldots, B_{21}B_{11}^i B_{12}$. Our induction hypothesis assumes that for $i \geq 1$

$$B^i = \begin{array}{c} \\ r \text{ rows} \\ \\ n-r \text{ rows} \end{array} \overset{\overset{r \text{ columns}}{\overbrace{}}}{\left(\begin{array}{cc} \overbrace{B_{11}^i + \sum_{j=0}^{i-2} B_{11}^j B_{12}B_{21}B_{11}^{i-2-j}}^{} & B_{11}^{i-1}B_{12} \\ B_{21}B_{11}^{i-1} & \underbrace{\mathbf{0}}_{n-r \text{ columns}} \end{array}\right)} \tag{5.3}$$

and $B_{22} = B_{21}B_{12} = B_{21}B_{11}B_{12} = \ldots = B_{21}B_{11}^{i-2}B_{12} = \mathbf{0}$. We just proved the base case of $i = 1$. Consider the following evaluation of $B^{i+1} = B \cdot B^i$

$$B^{i+1} = \begin{array}{c} \\ r \text{ rows} \\ n-r \text{ rows} \end{array} \overset{\overset{r \text{ columns}}{\overbrace{}}}{\left(\begin{array}{cc} B_{11}^{i+1} + \sum_{j=0}^{i-2} B_{11}^{j+1}B_{12}B_{21}B_{11}^{i-2-j} + B_{12}B_{21}B_{11}^{i-1} & B_{11}^i B_{12} \\ B_{21}B_{11}^i + \sum_{j=0}^{i-2} B_{21}B_{11}^j B_{12}B_{21}B_{11}^{i-2-j} & \underbrace{B_{21}B_{11}^{i-1}B_{12}}_{n-r \text{ columns}} \end{array}\right)}$$

(5.4)

Since $i+1 \leq k$, we must have $B_{21}B_{11}^{i-1}B_{12} = \mathbf{0}$, otherwise we would have $W_{i+1} \not\subseteq \mathrm{Im}(A)$. Also we know by the induction hypothesis that $B_{22} = B_{21}B_{12} = B_{21}B_{11}B_{12} = \ldots = B_{21}B_{11}^{i-2}B_{12} = \mathbf{0}$, this implies that

$$B^{i+1} = B \cdot B^i = \begin{matrix} \\ r \text{ rows} \\ \\ \\ \end{matrix} \overbrace{\begin{matrix} \left(\begin{matrix} \overbrace{B_{11}^{i+1} + \sum_{j=0}^{i-1} B_{11}^j B_{12} B_{21} B_{11}^{i-1-j}}^{r \text{ columns}} & \underbrace{B_{11}^i B_{12}}_{n-r \text{ columns}} \\ B_{21}B_{11}^i & \mathbf{0} \end{matrix} \right) \end{matrix}}^{} \begin{matrix} \} n-r \text{ rows} \\ \\ \\ \end{matrix} \tag{5.5}$$

Now we show that $k \geq \ell$. Since $k = w(A, \langle A, B\rangle)$, for all $1 \leq i \leq k$, B^i can be written as

$$B^i = \begin{matrix} \\ r \text{ rows} \\ \\ \\ \end{matrix} \overbrace{\begin{matrix} \left(\begin{matrix} \overbrace{B_{11}^i + \sum_{j=0}^{i-2} B_{11}^j B_{12} B_{21} B_{11}^{i-2-j}}^{r \text{ columns}} & \underbrace{B_{11}^{i-1} B_{12}}_{n-r \text{ columns}} \\ B_{21}B_{11}^{i-1} & \mathbf{0} \end{matrix} \right) \end{matrix}}^{} \begin{matrix} \} n-r \text{ rows} \\ \\ \\ \end{matrix} \tag{5.6}$$

Note that $\langle A, B\rangle^i$ is spanned by all matrices of the form $M_1M_2\cdots M_i$ with $M_j = A$ or $M_j = B$, $1 \leq j \leq i$. Since we have that $W_k \subseteq \mathrm{Im}(A)$, we know that $M_1M_2\cdots M_k \ker(A) \subseteq \mathrm{Im}(A)$ for any product $M_1M_2\cdots M_k$ as above. Now let us see what condition one needs such that $W_{k+1} \not\subseteq \mathrm{Im}(A)$ is true. Since A is the identity on $\mathrm{Im}(A)$, only B^{k+1} can take $\ker(A)$ out of $\mathrm{Im}(A)$ for $W_{k+1} \not\subseteq \mathrm{Im}(A)$ to be true. By a similar argument as above, this happens only when $B_{21}B_{11}^{k-1}B_{12} \neq \mathbf{0}$, thus $\ell \leq k$. ◀

Now, having established the connection between Wong index and the sequence of vanishing matrix products, we prove another technical lemma establishing the relation between the length of this sequence and the commutative rank.

▶ **Lemma 18.** *Let $B \in \mathbb{F}^{n\times n}$ and*

$$B = \begin{matrix} \\ r \text{ rows} \\ \\ \\ \end{matrix} \overbrace{\begin{matrix} \left(\begin{matrix} \overbrace{B_{11}}^{r \text{ columns}} & \underbrace{B_{12}}_{n-r \text{ columns}} \\ B_{21} & B_{22} \end{matrix} \right) \end{matrix}}^{} \begin{matrix} \} n-r \text{ rows} \\ \\ \\ \end{matrix} \tag{5.7}$$

Consider the sequence of matrices $B_{22}, B_{21}B_{12}, B_{21}B_{11}B_{12}, \ldots, B_{21}B_{11}^jB_{12}\ldots$. If the first $k \geq 1$ elements in this sequence are equal to the zero matrix and B_{11} is non-singular, then $\mathrm{rank}(B) \leq r\left(1 + \frac{1}{k}\right)$.

Proof. If $\mathrm{rank}(B_{12}) \leq \frac{r}{k}$, then we are done by using the Fact 15. So we can assume without loss of generality that $\mathrm{rank}(B_{12}) > \frac{r}{k}$. Now suppose that

$$\dim\langle \mathrm{Im}(B_{12}) \cup \mathrm{Im}(B_{11}B_{12}) \cup \ldots \cup \mathrm{Im}(B_{11}^{k-2}B_{12})\rangle \geq (k-1)\,\mathrm{rank}(B_{12}).$$

We note that $\mathrm{Im}(B_{12}), \mathrm{Im}(B_{11}B_{12}), \ldots, \mathrm{Im}(B_{11}^{k-2}B_{12})$, are sub-spaces of $\ker(B_{21})$. Further using the rank nullity theorem, we get $\mathrm{rank}(B_{21}) < r - \frac{r\cdot(k-1)}{k} = \frac{r}{k}$. By using Fact 15, we again get that $\mathrm{rank}(B) \leq r\left(1 + \frac{1}{k}\right)$.

In the above discussion, we assumed that

$$\dim\langle \mathrm{Im}(B_{12}) \cup \mathrm{Im}(B_{11}B_{12}) \cup \ldots \cup \mathrm{Im}(B_{11}^{k-2}B_{12})\rangle \geq (k-1)\,\mathrm{rank}(B_{12}).$$

What if this is not the case? We still want to use the same idea as above but we want to ensure this assumption. For this purpose, we use a series of elementary column operations on B to transform it to a new matrix B^*, which would satisfy above assumption. Since the rank of a matrix is invariant under elementary column operations, we would obtain the desired rank bound. Now we show how to obtain this matrix B^* using a series of elementary column operations on B. Whenever we apply these elementary column operations on B, we shall also maintain the invariant that $B_{22} = B_{21}B_{12} = B_{21}B_{11}B_{12} = \ldots = B_{21}B_{11}^{k-2}B_{12} = 0$.

Suppose

$$\dim\langle \operatorname{Im}(B_{12}) \cup \operatorname{Im}(B_{11}B_{12}) \cup \ldots \cup \operatorname{Im}(B_{11}^{k-2}B_{12})\rangle < (k-1)\operatorname{rank}(B_{12}). \tag{5.8}$$

Let $\rho := \operatorname{rank}(B_{12})$. First, we can assume that B_{12} has exactly ρ non-zero columns. This can be achieved by performing elementary column operations on the last $n-r$ columns. This does not change the matrix $B_{22} = 0$. Furthermore, these column operations correspond to replacing B_{12} by $B_{12} \cdot S$ for some invertible $(n-r) \times (n-r)$-matrix S. Since $B_{22} = B_{21}B_{12} = B_{21}B_{11}B_{12} = \ldots = B_{21}B_{11}^{k-2}B_{12} = 0$ implies $B_{21}B_{12}S = B_{21}B_{11}B_{12}S = \ldots = B_{21}B_{11}^{k-2}B_{12}S = 0$, we keep our invariant. We will call the new matrix again B_{12}.

Note that the image of a matrix is its column span. Since every matrix $B_{11}^iB_{12}$ has at most ρ non-zero columns (since B_{12} has ρ non-zero columns and B_{11} is non-singular), assumption 5.8 means that there is a linear dependence between these columns. That means there vectors $y_0, y_1, \ldots, y_{k-2} \in \mathbb{F}^{n-r}$, not all equal to zero, such that $\sum_{i=0}^{k-2} B_{11}^iB_{12} \cdot y_i = 0$. Moreover, these vectors only have non-zero entries in the places that corresponds to nonzero columns of B_{12}. First we show that we can assume $y_0 \neq 0$. Suppose $0 \leq j \leq k-2$ is the least integer such that $y_j \neq 0$. So we left multiply the equation $\sum_{i=0}^{k-2} B_{11}^iB_{12} \cdot y_i = 0$ by $(B_{11}^j)^{-1}$, giving us $(B_{11}^j)^{-1}\sum_{i=0}^{k-2} B_{11}^iB_{12} \cdot y_i = \sum_{i=j}^{k-2} B_{11}^{i-j}B_{12} \cdot y_i = 0$. By renumbering the indices, this can be re-written as $\sum_{i=0}^{k-2-j} B_{11}^iB_{12} \cdot y_i = 0$. Thus we can assume that $y_0 \neq 0$. (The new sum runs only up to $k-2-j$, for the missing summands, we choose the corresponding y_i to be zero.)

By writing $\sum_{i=0}^{k-2} B_{11}^iB_{12} \cdot y_i = 0$ as $B_{12} \cdot y_0 + B_{11} \cdot \sum_{i=1}^{k-2} B_{11}^{i-1}B_{12}y_i = 0$, we see that there is a linear dependence between the columns of B_{12} and B_{11}. Let $k \in [n-r]$ be such that k^{th} entry of y_0 is non-zero. Therefore, we can make the k^{th} column of B_{12} zero by adding a multiple of $\sum_{i=1}^{k-2} B_{11}^iB_{12} \cdot y_i$ and maybe adding some multiple of some other columns of B_{12} to it. This will decrease the rank of B_{12} by 1.

We claim that our invariant is still fulfilled. First, we add $B_{11} \cdot \sum_{i=1}^{k-2} B_{11}^{i-1}B_{12} \cdot y_i$ to the k^{th} column of B_{12} and this will also add $B_{21} \cdot \sum_{i=1}^{k-2} B_{11}^{i-1}B_{12} \cdot y_i$ to the k^{th} column of B_{22}. Since the invariant was fulfilled before the operation, B_{22} will stay zero. As seen before, column operations within the last $n-r$ columns do not change B_{22}. Thus, one of the $n-r$ columns on the right-hand side (side composed of B_{12} and B_{22}) of B became zero. We can remove this column from our consideration. Let B' and B'_{12} the matrices obtained from B and B_{12} by removing this zero column. Since the columns of B'_{12} are a subset of the columns of B_{12}, $B_{21}B_{12} = B_{21}B_{11}B_{12} = \ldots = B_{21}B_{11}^{k-2}B_{12} = 0$ implies that $B_{21}B'_{12} = B_{21}B_{11}B'_{12} = \ldots = B_{21}B_{11}^{k-2}B'_{12} = 0$. Therefore, our invariant is still valid.

We repeat this process until (5.8) is not true anymore. Note that this happens for sure when $\operatorname{rank}(B_{12}) = 0$. At the end of this process we get a matrix B^* such that

$$\dim\langle \operatorname{Im}(B^*_{12}) \cup \operatorname{Im}(B_{11}B^*_{12}) \cup \ldots \cup \operatorname{Im}(B_{11}^{k-2}B^*_{12})\rangle \geq (k-1)\operatorname{rank}(B^*_{12}).$$

Now the rank bound follows from the argument given above. ◀

Finally, combining the above three lemmas, the following lemma gives the desired quantitative relation between the commutative rank and Wong index, essential to the

analysis of our algorithm. It shows that higher the Wong index of the given matrix, the better it approximates the rank of the space.

▶ **Lemma 19.** *If* $A \in \mathcal{B} = \langle B_1, B_2, , \ldots, B_m \rangle \leq \mathbb{F}^{n\times n}$ *and* $B = \sum_{i=1}^{m} x_i B_i$, *then*

$$\operatorname{rank}(\mathcal{B}) = \operatorname{rank}(\langle A, B\rangle) \leq \operatorname{rank}(A)\left(1 + \frac{1}{w(A, \langle A, B\rangle)}\right). \tag{5.9}$$

Proof. Let $\operatorname{rank}(A) = r$. We use $\mathcal{C}$ to denote the matrix space $\langle A, B\rangle$, note that this space is being considered over the rational function field $\mathbb{F}(x_1, x_2, \ldots, x_m)$.

We know that there exist matrices $P, Q \in \mathbb{F}^{n\times n}$ such that

$$PAQ = \begin{bmatrix} I_r & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}. \tag{5.10}$$

Notice that $\operatorname{Im}(PAQ) = P\operatorname{Im}(A)$. Thus by Lemma 16, $w(A, \mathcal{C}) = w(PAQ, P\mathcal{C}Q)$. Also, it is easy to see that $\operatorname{rank}(A) = \operatorname{rank}(PAQ)$ and $\operatorname{rank}(\mathcal{C}) = \operatorname{rank}(P\mathcal{C}Q)$. Hence it is enough to show that

$$\operatorname{rank}(P\mathcal{C}Q) \leq \operatorname{rank}(PAQ)\left(1 + \frac{1}{w(PAQ, P\mathcal{C}Q)}\right). \tag{5.11}$$

For sake of simplicity, we just write $P\mathcal{C}Q$ as $\mathcal{C}$ and PAQ as A. Thus we have

$$A = \begin{bmatrix} I_r & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}. \tag{5.12}$$

We write B as

$$B = \begin{array}{c} \overbrace{}^{r \text{ columns}} \\ r \text{ rows } \{ \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix} \} n-r \text{ rows} \\ \underbrace{}_{n-r \text{ columns}} \end{array} \tag{5.13}$$

We get that B_{11} is non-singular over the field $\mathbb{F}(x_1, x_2, \ldots, x_m)$ since $A \in \mathcal{B}$. Also, we get by Lemma 17 that first $w(A, \mathcal{C})$ entries of the sequence of matrices $B_{22}, B_{21}B_{12}$, $B_{21}B_{11}B_{12}, \ldots, B_{21}B_{11}^{i}B_{12}\ldots$ are zero matrices. Now we apply lemma 18 to obtain that

$$\operatorname{rank}(B) = \operatorname{rank}(\mathcal{B}) = \operatorname{rank}(\mathcal{C}) \leq \operatorname{rank}(A)\left(1 + \frac{1}{w(A, \mathcal{C})}\right). \tag{5.14}$$

◀

▶ **Lemma 20.** *If* $A \in \mathcal{B} = \langle B_1, B_2, \ldots, B_m \rangle \leq \mathbb{F}^{n\times n}$, $B = \sum_{i=1}^{m} x_i B_i$ *and* $w(A, \langle A, B\rangle) < k$ for some $k \in [n]$, then there exist $1 \leq i_1, i_2, \ldots i_k \leq m$ and $\lambda_1, \lambda_2, \ldots, \lambda_k \in \mathbb{F}$ such that $w(A, \langle A, C\rangle) < k$, where $C = \lambda_1 B_{i_1} + \lambda_2 B_{i_2} + \ldots + \lambda_k B_{i_k}$.

Proof. Let $\operatorname{rank}(A) = r$. We know that there exist matrices $P, Q \in \mathbb{F}^{n\times n}$ such that

$$PAQ = \begin{bmatrix} I_r & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}. \tag{5.15}$$

Let $A' = PAQ$, $\mathcal{B}' = P\mathcal{B}Q$ and $B' = \sum_{i=1}^{m} x_i PB_iQ$. We write B' as

$$B' = \begin{array}{c} \overbrace{}^{r \text{ columns}} \\ r \text{ rows } \{ \begin{pmatrix} B'_{11} & B'_{12} \\ B'_{21} & B'_{22} \end{pmatrix} \} n-r \text{ rows} \\ \underbrace{}_{n-r \text{ columns}} \end{array} \tag{5.16}$$

By using Lemma 16, we know that $w(A, \langle A, B\rangle) = w(A', \langle A', B'\rangle) < k$. By using Lemma 17, we get that there exists $t \leq k$ such that $B'_{21}(B'_{11})^{t-2}B'_{12} \neq \mathbf{0}$ and

$$(B')^t = \begin{array}{c} \\ r \text{ rows} \\ \\ \\ \end{array} \overbrace{\begin{array}{cc} B''_{11} & B''_{12} \\ B''_{21} & B'_{21}\underbrace{(B'_{11})^{t-2}}B'_{12} \end{array}}^{r \text{ columns}} \Bigg\} n-r \text{ rows} \quad \begin{array}{c} \\ \\ \\ n-r \text{ columns} \end{array} \tag{5.17}$$

for some matrices $B''_{11}, B''_{12}, B''_{21}$. Since the entries of the matrix $B'_{21}(B'_{11})^{t-2}B'_{12}$ are polynomials in the variables $x_1, x_2, \ldots, x_m$ of degree at most k, there exists an assignment to these variables by field constants, assigning at most k variables non-zero values such that $B'_{21}(B'_{11})^{t-2}B'_{12}$ evaluates to a non-zero matrix. By using Lemma 17 again, this assignment gives us a matrix $C' \in \mathcal{B}'$ such that $w(A', \langle A', C'\rangle) < k$. By using Lemma 16, same assignment of the variables gives us a matrix $C \in \mathcal{B}$ such that $w(A, \langle A, C\rangle) < k$. ◀

6 Final Algorithm

Suppose we have a matrix space $\mathcal{B} = \langle B_1, B_2, , \ldots, B_m\rangle \leq \mathbb{F}^{n\times n}$, $B = \sum_{i=1}^m x_i B_i$ and a matrix $A \in \mathcal{B}$. Our goal is find a matrix D in $\mathcal{B}$ such that its rank is "close" to the commutative rank of $\mathcal{B}$. If the Wong index $w(A, \langle A, B\rangle)$ of A in $\langle A, B\rangle$ is "large", then we know by Lemma 19 that rank of of A is "close" to the commutative rank of $\mathcal{B}$, which is equal to the commutative rank of $\langle A, B\rangle$. What if this Wong index $w(A, \langle A, B\rangle)$ is "small"? Then we know that by Lemma 20 that by trying out small number (that means, $m^{w(A,\mathcal{B})+1}$) of possibilities of combinations of B_i, we can find a matrix $C \in \mathcal{B}$ such that Wong index $w(A, \langle A, B\rangle)$ of A in $\langle A, C\rangle$ is also "small". Using Fact 14, we obtain that rank of A is not maximum in $\langle A, C\rangle$. Thus there exists $\lambda \in \mathbb{F}$ such that $\operatorname{rank}(A + \lambda C) > \operatorname{rank}(A)$. And we can find this λ quite efficiently. Also, $A + \lambda C \in \mathcal{B}$. Thus we can efficiently find a matrix of bigger rank if we are given a matrix of "small" Wong index. This idea is formalized in the following Algorithm.

Input : A matrix space $\mathcal{B} = \langle B_1, B_2, \ldots, B_m\rangle \leq \mathbb{F}^{n\times n}$, given as a list of basis matrices $B_1, B_2, \ldots, B_m$. An approximation parameter $0 < \epsilon < 1$.
Output : A matrix $A \in \mathcal{B}$ such that $\operatorname{rank}(A) \geq (1-\epsilon)\cdot \operatorname{rank}(\mathcal{B})$
Initialize $A = 0 \in \mathbb{F}^{n\times n}$ to the zero matrix.
Assign $\ell = \lceil \frac{1}{\epsilon} - 1\rceil$.
while *Rank is increasing* **do**
  for *each* $\{i_1, i_2, \ldots, i_\ell\} \in \binom{[m]\setminus\{0\}}{\ell}$ **do**
    `/* This means we try all combinations of matrices` $B_{i_1}, B_{i_2}, \ldots, B_{i_\ell}$ `*/`
    Check if there exist $\lambda_1, \lambda_2, \ldots, \lambda_\ell \in \mathbb{F}$ such that $\operatorname{rank}(A+\lambda_1 B_{i_1}+\lambda_2 B_{i_2}+\ldots+\lambda_k B_{i_\ell}) > \operatorname{rank}(A)$. **if** $\operatorname{rank}(A + \lambda_1 B_{i_1} + \lambda_2 B_{i_2} + \ldots + \lambda_k B_{i_\ell}) > \operatorname{rank}(A)$ **then**
      Update $A = A + \lambda_1 B_{i_1} + \lambda_2 B_{i_2} + \ldots + \lambda_k B_{i_\ell}$.
return A.

Algorithm 2 Greedy algorithm for $(1-\epsilon)$-approximating commutative rank.

The following theorem proves the correctness of Algorithm 2. Let s be an upper bound on the bit size of the entries of $B_1, \ldots, B_m$.

▶ **Theorem 21.** *Assume that $|\mathbb{F}| > n$. Algorithm 2 runs in time $O((mn)^{\frac{1}{\epsilon}} \cdot M(n, s+\log n) \cdot n)$ and returns a matrix $A \in \mathcal{B}$ such that $\operatorname{rank}(A) \geq (1-\epsilon) \cdot \operatorname{rank}(\mathcal{B})$, where $M(n,t)$ is the time required to compute the rank of an $n \times n$ matrix with entries of bit size at most t.*

Proof. Suppose $B = \sum_{i=1}^{m} x_i B_i$ and A be the rank r matrix returned by Algorithm 2. Let k be the Wong index $w(A, \langle A, B\rangle)$ of $(A, \langle A, B\rangle)$. By Lemma 19, we know that $\operatorname{rank}(\mathcal{B}) \leq r\left(1 + \frac{1}{k}\right)$. Thus $r \geq (1 - \frac{1}{k+1}) \operatorname{rank}(\mathcal{B})$. If $\epsilon \geq \frac{1}{k+1}$, then we are done. Otherwise we have that $\epsilon < \frac{1}{k+1}$, i.e, $k < \frac{1}{\epsilon} - 1$. Since $\ell = \lceil \frac{1}{\epsilon} - 1 \rceil$, we also have $w(A, \langle A, B\rangle) < \ell$. By using Lemma 20, we get that there exist $1 \leq i_1, i_2, \ldots i_\ell \leq m$ and $\lambda_1, \lambda_2, \ldots, \lambda_\ell \in \mathbb{F}$ such that that $w(A, \langle A, C\rangle) < \ell$, where $C = \lambda_1 B_{i_1} + \lambda_2 B_{i_2} + \ldots + \lambda_\ell B_{i_\ell}$. By using Fact 14, we get that A is not of maximum rank in $\langle A, C\rangle$. Thus there exists $\lambda \in \mathbb{F}$ such that $\operatorname{rank}(A + \lambda C) > A$, and we shall detect this in Algorithm 2 since we try all possible choices of $i_1, i_2, \ldots, i_\ell$.

The desired running time can be proved easily. The outer while loop runs at most n times, thus the total running time is at most n times the running time of one iteration. One iteration of the outer loop has $\binom{[m]\setminus\{0\}}{\ell} = O(m^{\frac{1}{\epsilon}})$ iterations of the inner for loop. By using the Schwartz–Zippel Lemma [22, 18], one iteration of inner for loop needs to try at most $(n+1)^\ell = O(n^{\frac{1}{\epsilon}})$ possible values of $\lambda_1, \lambda_2, \ldots, \lambda_\ell \in \mathbb{F}$. And then we perform two instances of rank computation. The stated running time follows. ◀

▶ Remark. Algorithm 2 runs in time $O((mn)^{\frac{1}{\epsilon}} \cdot n \cdot M(n))$ in the algebraic RAM model. Here $M(n)$ is the time required to compute the rank of an $n \times n$ matrix in the algebraic RAM model. It is known that $M(n) = O(n^\omega)$ with ω being the exponent of matrix multiplication. Since one can assume that $m \leq n^2$, Algorithm 2 runs in time $O(n^{\frac{3}{\epsilon}+\omega+1})$ in algebraic ram model.

The statement of the above remark and the trivial fact that $\omega \leq 3$, gives us the running time stated in the abstract.

▶ Remark. With a more refined analysis, it can be seen that Algorithm 2 uses $O((mn)^{\frac{1}{\epsilon}} \cdot n \cdot M(n, s + \log n))$ bit operations if the entries of the input matrices $B_1, B_2, \ldots, B_m$ have bit size at most s. Here $M(n,t)$ is the bit complexity of computing the rank of a matrix whose entries have bit size at most t. The additional $\log n$ in the bit size comes from the fact that the entries of the final matrix A are by a polynomial factor (in n) larger than the entries of the B_i due to the update steps.

7 Tight examples

We conclude by giving some tight examples, which show that the analysis of the approximation performance of the greedy approximation scheme cannot be improved. Consider the following matrix space of $n \times n$-matrices:

$$\left(\begin{array}{cccc|cccc} * & 0 & \ldots & 0 & * & 0 & \ldots & 0 \\ 0 & * & \ldots & 0 & 0 & * & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & * & 0 & 0 & \ldots & * \\ \hline 0 & 0 & \ldots & 0 & * & 0 & \ldots & 0 \\ 0 & 0 & \ldots & 0 & 0 & * & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & 0 & 0 & 0 & \ldots & * \end{array}\right) \tag{7.1}$$

Each block has size $\frac{n}{2} \times \frac{n}{2}$. This space consists of all matrices where we can substitute arbitrary values for the $*$ and the basis consists of all matrices where exactly one $*$ is replaced

by 1 and all others are set to 0. Assume that $\epsilon = \frac{1}{2}$, that means, that the greedy algorithm only looks at sets of size $\ell = 1$. Furthermore, assume that the matrix A constructed so far is

$$A = \begin{pmatrix} \mathbf{0} & I_{\frac{n}{2}} \\ \mathbf{0} & \mathbf{0} \end{pmatrix}. \tag{7.2}$$

Any single basis matrix cannot improve the rank of A, since either its nonzero column is contained in the column span of A or its nonzero row is contained in the row span of A. On the other hand, the matrix space contains a matrix of full rank n, namely, the identity matrix.

The next space for the case $\ell = 2$ looks like this:

$$\left(\begin{array}{cccc|cccc|cccc} * & 0 & \dots & 0 & * & 0 & \dots & 0 & 0 & 0 & \dots & 0 \\ 0 & * & \dots & 0 & 0 & * & \dots & 0 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & * & 0 & 0 & \dots & * & 0 & 0 & \dots & 0 \\ \hline 0 & 0 & \dots & 0 & * & 0 & \dots & 0 & * & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & 0 & * & \dots & 0 & 0 & * & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & * & 0 & 0 & \dots & * \\ \hline 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & * & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & * & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 0 & \dots & * \end{array}\right) \tag{7.3}$$

and the corresponding matrix A is

$$A = \begin{pmatrix} \mathbf{0} & I_{\frac{2n}{3}} \\ \mathbf{0} & \mathbf{0} \end{pmatrix}. \tag{7.4}$$

By an argument similar to above, it is easy to see that we need at least three matrices to improve the rank of A, so the algorithm gets stuck with a $\frac{2}{3}$-approximation.

The above scheme generalizes to arbitrary values of ℓ in the obvious way.

Acknowledgments. We would like to thank Ankit Garg and Rafael Oliveira for pointing out some inaccuracies in an earlier draft of the paper. We would also like to thank the anonymous reviewers for their helpful and constructive comments that have helped us to improve the final version of the paper. We also thank the editors for their support during the review process.

References

1 Jonathan F. Buss, Gudmund S. Frandsen, and Jeffrey O. Shallit. The computational complexity of some problems of linear algebra. *Journal of Computer and System Sciences*, 58(3):572–596, 1999.

2 Harm Derksen and Visu Makam. On non-commutative rank and tensor rank. *arXiv preprint arXiv:1606.06701*, 2016.

3 Jack Edmonds. Systems of distinct representatives and linear algebra. *J. Res. Nat. Bur. Standards Sect. B*, 71:241–245, 1967.

4 David Eisenbud and Joe Harris. Vector spaces of matrices of low rank. *Advances in Mathematics*, 70(2):135–155, 1988.

5 Marc Fortin and Christophe Reutenauer. Commutative/noncommutative rank of linear matrices and subspaces of matrices of low rank. *Séminaire Lotharingien de Combinatoire*, 52:B52f, 2004. URL: `http://eudml.org/doc/125000`.

6 Harold N. Gabow and Matthias Stallmann. An augmenting path algorithm for linear matroid parity. *Combinatorica*, 6(2):123–150, 1986.

7 Ankit Garg, Leonid Gurvits, Rafael Oliveira, and Avi Wigderson. A deterministic polynomial time algorithm for non-commutative rational identity testing. *CoRR*, abs/1511.03730, 2015. URL: `http://arxiv.org/abs/1511.03730`.

8 Leonid Gurvits. Classical complexity and quantum entanglement. *Journal of Computer and System Sciences*, 69(3):448–484, 2004.

9 Gábor Ivanyos, Marek Karpinski, Youming Qiao, and Miklos Santha. Generalized Wong sequences and their applications to Edmonds' problems. *Journal of Computer and System Sciences*, 81(7):1373–1386, 2015. URL: `http://arxiv.org/abs/1307.6429`.

10 Gábor Ivanyos, Marek Karpinski, and Nitin Saxena. Deterministic polynomial time algorithms for matrix completion problems. *SIAM Journal on Computing*, 39(8):3736–3751, 2010.

11 Gábor Ivanyos, Youming Qiao, and K. V. Subrahmanyam. Constructive noncommutative rank computation in deterministic polynomial time over fields of arbitrary characteristics. *arXiv preprint arXiv:1512.03531*, 2015.

12 László Lovász. The matroid matching problem. *Algebraic Methods in Graph Theory*, 1:495–517, 1978.

13 László Lovász. On determinants, matchings, and random algorithms. In *FCT*, volume 79 of *LNCS*, pages 565–574, 1979.

14 László Lovász. Singular spaces of matrices and their application in combinatorics. *Boletim da Sociedade Brasileira de Matemática-Bulletin/Brazilian Mathematical Society*, 20(1):87–99, 1989.

15 László Lovász and Michael D. Plummer. *Matching theory*, volume 367. American Mathematical Soc., 2009.

16 Meena Mahajan and V. Vinay. Determinant: Combinatorics, algorithms, and complexity. *Chicago Journal of Theoretical Computer Science*, 1997:5, 1997.

17 James B. Orlin. A fast, simpler algorithm for the matroid parity problem. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 240–258. Springer, 2008.

18 Jacob T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *Journal of the ACM*, 27(4):701–717, 1980.

19 Seinosuke Toda. Counting problems computationally equivalent to computing the determinant. *Technical Report CSIM 91-07*, 1991.

20 L. G. Valiant. Completeness classes in algebra. In *Proceedings of the Eleventh Annual ACM Symposium on Theory of Computing*, STOC'79, pages 249–261, New York, NY, USA, 1979. ACM. `doi:10.1145/800135.804419`.

21 V. Vinay. Counting auxiliary pushdown automata and semi-unbounded arithmetic circuits. In *Structure in Complexity Theory Conference, 1991, Proceedings of the Sixth Annual*, pages 270–284. IEEE, 1991.

22 Richard Zippel. Probabilistic algorithms for sparse polynomials. In Edward W. Ng, editor, *EUROSAM*, volume 72 of *Lecture Notes in Computer Science*, pages 216–226. Springer, 1979. URL: `http://dblp.uni-trier.de/db/conf/eurosam/eurosam1979.html#Zippel79`, `doi:10.1007/3-540-09519-5_73`.

A Appendix

Here we present some proofs which were omitted in the main manuscript.

▶ **Lemma 22.** *For all matrix spaces* $\mathcal{B} \leq \mathbb{F}^{n\times n}$, $\operatorname{rank}(\mathcal{B}) \leq \text{nc-rank}(\mathcal{B})$.

Proof. Let $r = \text{nc-rank}(\mathcal{B})$. This means that there exists $V \leq \mathbb{F}^n$ such that $\operatorname{rank}(\mathcal{B}V) = \dim(V) - (n-r)$. Therefore, for all $B \in \mathcal{B}$, $\operatorname{rank}(BV) \leq \dim(V) - (n-r)$. Thus $\operatorname{rank}(\mathcal{B}) \leq n - (n-r) = r = \text{nc-rank}(\mathcal{B})$. ◀

▶ **Lemma 23.** *Algorithm 1 runs in polynomial time and returns a matrix* $A \in \mathcal{B}$ *such that* $\operatorname{rank}(A) \geq \frac{1}{2} \cdot \operatorname{rank}(\mathcal{B})$.

Proof. Let A be the matrix returned by Algorithm 1. Assume that A has rank r. We know that there exist non-singular matrices P and Q such that

$$PAQ = \begin{pmatrix} I_r & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 \end{pmatrix}, \tag{A.1}$$

where I_r is the $r \times r$ identity matrix. Now consider the matrix space $P\mathcal{B}Q := \langle PB_1Q, PB_2Q, \dots, PB_mQ\rangle$. This does not change anything with respect to the rank. So for the analysis, we can replace $\mathcal{B}$ by $P\mathcal{B}Q$. Consider any general matrix $A + x_1B_1 + x_2B_2 + \dots + x_mB_m$ in $\mathcal{B}$. We decompose it as

$$A + x_1B_1 + x_2B_2 + \dots + x_mB_m = \left(\begin{array}{c|c} M_1 & M_2 \\ \hline M_3 & M_4 \end{array}\right). \tag{A.2}$$

Here M_1 is an $r \times r$ matrix, M_2 is an $r \times (n-r)$ matrix, M_3 is a $(n-r) \times r$ matrix and M_4 is a $(n-r) \times (n-r)$ matrix. M_1, M_2, M_3, and M_4 have (affine) linear forms in variables $\mathbf{x} = (x_1, x_2, \dots, x_m)$ as their entries.

Now we claim that the bottom right part M_4 is the zero matrix. Assume otherwise. Assume that the (s,t)-entry of the above matrix is nonzero with $s, t > r$. Consider the $(r+1) \times (r+1)$ minor of $A + x_1B_1 + x_2B_2 + \dots + x_mB_m$, obtained by adding the sth row (from M_3) and the tth column (from M_2) to M_1. We shall denote this minor by C. The minor C looks like

$$C = \begin{pmatrix} 1 + \ell_{11}(\mathbf{x}) & \ell_{12}(\mathbf{x}) & \dots & \ell_{1r}(\mathbf{x}) & a_1(\mathbf{x}) \\ \ell_{21}(\mathbf{x}) & 1 + \ell_{22}(\mathbf{x}) & \dots & \ell_{2r}(\mathbf{x}) & a_2(\mathbf{x}) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \ell_{r1}(\mathbf{x}) & \ell_{r2}(\mathbf{x}) & \dots & 1 + \ell_{rr}(\mathbf{x}) & a_r(\mathbf{x}) \\ b_1(\mathbf{x}) & b_2(\mathbf{x}) & \dots & b_r(\mathbf{x}) & c(\mathbf{x}) \end{pmatrix}. \tag{A.3}$$

The $\ell_{i,j}$, a_i, b_j, and c are homogeneous linear forms in $\mathbf{x}$. By our choice, $c(\mathbf{x}) \neq 0$. It is not hard to see that

$$\det(C) = c(\mathbf{x}) + \text{terms of degree at least 2.} \tag{A.4}$$

Thus there are $\lambda \in \mathbb{F}$ and $i \in [m]$ such that $\det(C(\alpha)) \neq 0$, where α is the assignment to the variables $\mathbf{x} = (x_1, x_2, \dots, x_m)$ obtained by setting $x_k = 0$ when $k \neq i$ and $x_i = \lambda$. These choices of $i \in [m]$ and $\lambda \in \mathbb{F}$ would allow Algorithm 1 to find a matrix A of larger rank. Thus Algorithm 1 would keep finding a matrix A of larger rank when the matrix M_4 is non-zero. Hence it can only stop when M_4 is the zero matrix. If M_4 is the zero

matrix then $\text{rank}(\mathcal{B}) \leq 2r$. Thus when Algorithm 1 stops, it outputs a matrix A such that $\text{rank}(A) \geq \frac{1}{2} \cdot \text{rank}(\mathcal{B})$.

The running time is obviously polynomial since the while loop is executed at most n times and we have to check at most $n+1$ values for λ. The size of the numbers that occur in the rank check is polynomial in the size of the entries of $B_1, \ldots, B_m$. ◀

▶ **Lemma 24.** *Let $\mathcal{B} \leq \mathbb{F}^{n\times n}$ be a matrix space, $A \in \mathcal{B}$, A' be a pseudo-inverse of A and $(W_i)_{i\in[n]}$ be the second Wong sequence of $(A, \mathcal{B})$. Then for all $1 \leq i \leq n$, we have $W_i = (\mathcal{B}A')^i(\text{Ker}(AA'))$ as long as $W_{i-1} \subseteq \text{Im}\, A$.*

Proof. We prove the statement by induction on i. Since $\text{Ker}(AA') = A'^{-1}(\text{Ker}(A))$, we get that $(\mathcal{B}A')(\text{Ker}(AA')) = \mathcal{B}A'A'^{-1}(\text{Ker}(A)) = \mathcal{B}\,\text{Ker}(A) = W_1$. This proves the base case of $i = 1$. To prove that $W_i = (\mathcal{B}A')^i(\text{Ker}(AA'))$, we shall prove that $(\mathcal{B}A')^i(\text{Ker}(AA')) \subseteq W_i$ and $W_i \subseteq (\mathcal{B}A')^i(\text{Ker}(AA'))$. By the induction hypothesis, we just need to prove that $(\mathcal{B}A')(W_{i-1}) \subseteq W_i$ and $W_i \subseteq (\mathcal{B}A')(W_{i-1})$.

First we prove the easy direction, that is $(\mathcal{B}A')(W_{i-1}) \subseteq W_i$. Since $W_{i-1} \subseteq \text{Im}(A)$, we have that $A'(W_{i-1}) \subseteq A^{-1}(W_{i-1})$. Thus $(\mathcal{B}A')(W_{i-1}) \subseteq (\mathcal{B}A^{-1})(W_{i-1}) = W_i$.

Now we prove that $W_i \subseteq (\mathcal{B}A')(W_{i-1})$. Since $W_{i-1} \subseteq \text{Im}(A)$, we get that $A^{-1}(W_{i-1}) = A'W_{i-1} + \text{Ker}(A)$. Thus $W_i = \mathcal{B}A^{-1}(W_{i-1}) \subseteq \mathcal{B}A'W_{i-1} + \mathcal{B}\,\text{Ker}(A)$. We have $\mathcal{B}\,\text{Ker}(A) = W_1 \subseteq W_{i-1}$, this implies that $W_i \subseteq \mathcal{B}A'W_{i-1} + W_{i-1}$. Since $A \in \mathcal{B}$ and $W_{i-1} = AA'W_{i-1}$, we get that $W_{i-1} \subseteq \mathcal{B}A'W_{i-1}$. This in turn implies that $W_i \subseteq \mathcal{B}A'W_{i-1} + \mathcal{B}A'W_{i-1} = (\mathcal{B}A')(W_{i-1})$. ◀

▶ **Lemma 25.** *Let $A \in \mathcal{B} \leq \mathbb{F}^{n\times n}$ and $(W_i)_{i\in[n]}$ be the second Wong sequence of $(A, \mathcal{B})$. If $P \in \mathbb{F}^{n\times n}$ and $Q \in \mathbb{F}^{n\times n}$ are invertible matrices, then the second Wong sequence of $(PAQ, P\mathcal{B}Q)$ is $(PW_i)_{i\in[n]}$. In particular, $w(A, \mathcal{B}) = w(PAQ, P\mathcal{B}Q)$.*

Proof. Consider the ith entry W'_i in the second Wong sequence of $(PAQ, P\mathcal{B}Q)$. We prove that $W'_i = PW_i$ for all $i \in [n]$. We use induction on i. The statement is trivially true for $i = 0$. By the induction hypothesis, we have, $W'_i = P\mathcal{B}Q(PAQ)^{-1}PW_{i-1} = P\mathcal{B}QQ^{-1}A^{-1}P^{-1}PW_{i-1} = P\mathcal{B}A^{-1}(W_{i-1}) = PW_i$. ◀